
Contents

Editorial

Guest Editorial: Adaptive Smart Areas and Intelligent Agents
Guest Editorial: Role of Agents in Traffic and Transportation
Guest Editorial: Innovations in Intelligent Systems and Applications

Papers

- 1 FSASA: Sequential Recommendation Based on Fusing Session-Aware Models and Self-Attention Networks
Shangzhi Guo, Xiaofeng Liao, Fei Meng, Qing Zhao, Yuling Tang, Hui Li, Qinqin Zong
- 21 Applying SPIN Checker on 5G EAP-TLS Authentication Protocol Analysis
Qianli Wang
- 37 PI-BODE: Programmable Intraflow-based IoT Botnet Detection system
Đorđe D. Jovanović, Pavle V. Vuletić
- 57 Feature Parameters extraction and Affective Computing of Voice Message for Social Media Environment
Peng Jiang, Cui Guo, Yonghui Dai
- 75 Machine Learning and Text Mining based Real-Time Semi-Autonomous Staff Assignment System
ARSLAN, Yunus Emre IŞIK, Yasin GÖRMEZ, Mustafa TEMİZ
- 95 Activity Recognition for Elderly Care Using Genetic Search
Ankita Biswal, Chhabi Rani Panigrahi, Anukampa Behera, Sarmistha Nanda, Tien-Hsiung Weng, Bibudhendu Pati, Chandan Malu
- 117 Comparing Semantic Graph Representations of Source Code: The Case of Automatic Feedback on Programming Assignments
José Carlos Paiva, José Paulo Leal, Álvaro Figueira
- 143 MK-MSVCR: An Efficient Multiple Kernel Approach to Multi-class Classification
Zijie Dong, Fen Chen, Yu Zhang
- 167 An Approach for Supporting Transparent ACID Transactions over Heterogeneous Data Stores in Microservice Architectures
Lazar Nikolić, Vladimir Dimitrieski, Milan Čeliković
- 203 SPC5: an efficient SpMV framework vectorized using ARM SVE and x86 AVX-512
Evann Regnault, Bérénger Bramas

Special Section: Adaptive Smart Areas and Intelligent Agents

- 223 Evaluation of Deep Learning Techniques for Plant Disease Detection
Cedric Marco-Detchart, Jaime Andrés Rincon, Carlos Carrascosa, Vicente Julian
- 245 A Flexible Approach for Demand-Responsive Public Transport in Rural Areas
Pasqual Martí, Jaume Jordán, Vicente Julian
- 269 How to Fairly and Efficiently Assign Tasks in Individually Rational Agents' Coalitions? Models and Fairness Measures
Marin Lujak, Alessio Salvatore, Alberto Fernández, Stefano Giordani, Kendal Cousy

Special Section: Role of Agents in Traffic and Transportation

- 291 Comparing Reinforcement Learning Algorithms for a Trip Building Task: a Multi-objective Approach Using Non-Local Information
Henrique U. Gobbi, Guilherme Dytz dos Santos, Ana L. C. Bazzan
- 309 Sustainability-Oriented Route Generation for Ridesharing Services
Mengya Liu, Vahid Yazdanpanah, Sebastian Stein, Enrico Gerding
- 335 Knowledge Transfer in Multi-Objective Multi-Agent Reinforcement Learning via Generalized Policy Improvement
Vicente N. de Almeida, Lucas N. Alegre, Ana L. C. Bazzan

Special Section: Innovations in Intelligent Systems and Applications

- 363 3D Convolutional Long Short-Term Encoder-Decoder Network for Moving Object Segmentation
Anil Turker, Ender Mete Eksioğlu
- 379 Echo State Network for Features Extraction and Segmentation of Tomography Images
Petia Koprinkova-Hristova, Ivan Georgiev, Miryana Raykovska
- 395 VINIA: Voice-Enabled Intent-Based Networking for Industrial Automation
Raul Barbosa, João Fonseca, Marco Araújo, Daniel Corujo

Computer Science and Information Systems

Published by ComSIS Consortium

Volume 21, Number 1
January 2024

ComSIS is an international journal published by the ComSIS Consortium

ComSIS Consortium:

University of Belgrade:

Faculty of Organizational Science, Belgrade, Serbia
Faculty of Mathematics, Belgrade, Serbia
School of Electrical Engineering, Belgrade, Serbia

Serbian Academy of Science and Art:

Mathematical Institute, Belgrade, Serbia

Union University:

School of Computing, Belgrade, Serbia

University of Novi Sad:

Faculty of Sciences, Novi Sad, Serbia
Faculty of Technical Sciences, Novi Sad, Serbia
Technical Faculty "Mihajlo Pupin", Zrenjanin, Serbia

University of Niš:

Faculty of Electronic Engineering, Niš, Serbia

University of Montenegro:

Faculty of Economics, Podgorica, Montenegro

EDITORIAL BOARD:

Editor-in-Chief: Mirjana Ivanović, University of Novi Sad

Vice Editor-in-Chief: Boris Delibašić, University of Belgrade

Managing Editors:

Vladimir Kurbalija, University of Novi Sad
Miloš Radovanović, University of Novi Sad

Editorial Assistants:

Jovana Vidaković, University of Novi Sad
Ivan Pribela, University of Novi Sad
Davorka Radaković, University of Novi Sad
Slavica Kordić, University of Novi Sad
Srđan Škrbić, University of Novi Sad

Editorial Board:

A. Badica, *University of Craiova, Romania*
C. Badica, *University of Craiova, Romania*
M. Bajec, *University of Ljubljana, Slovenia*
L. Bellatreche, *ISAE-ENSMA, France*
I. Berković, *University of Novi Sad, Serbia*
D. Bojić, *University of Belgrade, Serbia*
Z. Bosnic, *University of Ljubljana, Slovenia*
D. Brđanin, *University of Banja Luka, Bosnia and Hercegovina*
R. Chbeir, *University Pau and Pays Adour, France*
M-Y. Chen, *National Cheng Kung University, Tainan, Taiwan*
C. Chesšev, *Universidad Nacional del Sur, Bahía Blanca, Argentina*
W. Dai, *Fudan University Shanghai, China*
P. Delias, *International Hellenic University, Kavala University, Greece*
B. Delibašić, *University of Belgrade, Serbia*
G. Devedžić, *University of Kragujevac, Serbia*
J. Eder, *Alpen-Adria-Universität Klagenfurt, Austria*
Y. Fan, *RMIT University, Australia*
V. Filipović, *University of Belgrade, Serbia*
T. Galinac Grbac, *Juraj Dobrića University of Pula, Croatia*
H. Gao, *Shanghai University, China*
M. Gušev, *Ss. Cyril and Methodius University Skopje, North Macedonia*
D. Han, *Shanghai Maritime University, China*
M. Heričko, *University of Maribor, Slovenia*
M. Holbl, *University of Maribor, Slovenia*
L. Jain, *University of Canberra, Australia*
D. Janković, *University of Niš, Serbia*
J. Janousek, *Czech Technical University, Czech Republic*
G. Jezic, *University of Zagreb, Croatia*
G. Kardas, *Ege University International Computer Institute, Izmir, Turkey*
Lj. Kaščelan, *University of Montenegro, Montenegro*
P. Keřalas, *City College, Thessaloniki, Greece*
M-K. Khan, *King Saud University, Saudi Arabia*
S-W. Kim, *Hanyang University, Seoul, Korea*
M. Kirikova, *Riga Technical University, Latvia*
A. Klašnja Milićević, *University of Novi Sad, Serbia*
J. Kratica, *Institute of Mathematics SANU, Serbia*
K-C. Li, *Providence University, Taiwan*
M. Lujak, *University Rey Juan Carlos, Madrid, Spain*
JM. Machado, *School of Engineering, University of Minho, Portugal*
Z. Maamar, *Zayed University, UAE*
Y. Manolopoulos, *Aristotle University of Thessaloniki, Greece*
M. Mernik, *University of Maribor, Slovenia*
B. Milašinović, *University of Zagreb, Croatia*
A. Mishev, *Ss. Cyril and Methodius University Skopje, North Macedonia*
N. Mitić, *University of Belgrade, Serbia*
N-T. Nguyen, *Wroclaw University of Science and Technology, Poland*
P Novais, *University of Minho, Portugal*
B. Novikov, *St Petersburg University, Russia*
M. Paprzický, *Polish Academy of Sciences, Poland*
P. Peris-Lopez, *University Carlos III of Madrid, Spain*
J. Protić, *University of Belgrade, Serbia*
M. Racković, *University of Novi Sad, Serbia*
M. Radovanović, *University of Novi Sad, Serbia*
P. Rajković, *University of Nis, Serbia*
O. Romero, *Universitat Politècnica de Catalunya, Barcelona, Spain*
C. Savaglio, *ICAR-CNR, Italy*
H. Shen, *Sun Yat-sen University, China*
J. Sierra, *Universidad Complutense de Madrid, Spain*
B. Stantic, *Griffith University, Australia*
H. Tian, *Griffith University, Australia*
N. Tomašev, *Google, London*
G. Trajčevski, *Northwestern University, Illinois, USA*
G. Velinov, *Ss. Cyril and Methodius University Skopje, North Macedonia*
L. Wang, *Nanyang Technological University, Singapore*
F. Xia, *Dalian University of Technology, China*
S. Xinogalos, *University of Macedonia, Thessaloniki, Greece*
S. Yin, *Software College, Shenyang Normal University, China*
K. Zdravkova, *Ss. Cyril and Methodius University Skopje, North Macedonia*
J. Zdravković, *Stockholm University, Sweden*

ComSIS Editorial Office:

**University of Novi Sad, Faculty of Sciences,
Department of Mathematics and Informatics**
Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia
Phone: +381 21 458 888; **Fax:** +381 21 6350 458
www.comsis.org; Email: comsis@uns.ac.rs

Volume 21, Number 1, 2024
Novi Sad

Computer Science and Information Systems

ISSN: 2406-1018 (Online)

The ComSIS journal is sponsored by:

Ministry of Education, Science and Technological Development of the Republic of Serbia
<http://www.mps.gov.rs/>



Computer Science and Information Systems

AIMS AND SCOPE

Computer Science and Information Systems (ComSIS) is an international refereed journal, published in Serbia. The objective of ComSIS is to communicate important research and development results in the areas of computer science, software engineering, and information systems.

We publish original papers of lasting value covering both theoretical foundations of computer science and commercial, industrial, or educational aspects that provide new insights into design and implementation of software and information systems. In addition to wide-scope regular issues, ComSIS also includes special issues covering specific topics in all areas of computer science and information systems.

ComSIS publishes invited and regular papers in English. Papers that pass a strict reviewing procedure are accepted for publishing. ComSIS is published semiannually.

Indexing Information

ComSIS is covered or selected for coverage in the following:

- Science Citation Index (also known as SciSearch) and Journal Citation Reports / Science Edition by Thomson Reuters, with 2022 two-year impact factor 1.4,
- Computer Science Bibliography, University of Trier (DBLP),
- EMBASE (Elsevier),
- Scopus (Elsevier),
- Summon (Serials Solutions),
- EBSCO bibliographic databases,
- IET bibliographic database Inspec,
- FIZ Karlsruhe bibliographic database io-port,
- Index of Information Systems Journals (Deakin University, Australia),
- Directory of Open Access Journals (DOAJ),
- Google Scholar,
- Journal Bibliometric Report of the Center for Evaluation in Education and Science (CEON/CEES) in cooperation with the National Library of Serbia, for the Serbian Ministry of Education and Science,
- Serbian Citation Index (SCIndeks),
- doiSerbia.

Information for Contributors

The Editors will be pleased to receive contributions from all parts of the world. An electronic version (LaTeX), or three hard-copies of the manuscript written in English, intended for publication and prepared as described in "Manuscript Requirements" (which may be downloaded from <http://www.comsis.org>), along with a cover letter containing the corresponding author's details should be sent to official journal e-mail.

Criteria for Acceptance

Criteria for acceptance will be appropriateness to the field of Journal, as described in the Aims and Scope, taking into account the merit of the content and presentation. The number of pages of submitted articles is limited to 20 (using the appropriate LaTeX template).

Manuscripts will be refereed in the manner customary with scientific journals before being accepted for publication.

Copyright and Use Agreement

All authors are requested to sign the "Transfer of Copyright" agreement before the paper may be published. The copyright transfer covers the exclusive rights to reproduce and distribute the paper, including reprints, photographic reproductions, microform, electronic form, or any other reproductions of similar nature and translations. Authors are responsible for obtaining from the copyright holder permission to reproduce the paper or any part of it, for which copyright exists.

Computer Science and Information Systems

Volume 21, Number 1, January 2024

CONTENTS

Editorial

Guest Editorial: Adaptive Smart Areas and Intelligent Agents

Guest Editorial: Role of Agents in Traffic and Transportation

Guest Editorial: Innovations in Intelligent Systems and Applications

Papers

- 1 FSASA: Sequential Recommendation Based on Fusing Session-Aware Models and Self-Attention Networks**
Shangzhi Guo, Xiaofeng Liao, Fei Meng, Qing Zhao, Yuling Tang, Hui Li, Qinqin Zong
- 21 Applying SPIN Checker on 5G EAP-TLS Authentication Protocol Analysis**
Qianli Wang
- 37 PI-BODE: Programmable Intraflow-based IoT Botnet Detection system**
Đorđe D. Jovanović, Pavle V. Vuletić
- 57 Feature Parameters extraction and Affective Computing of Voice Message for Social Media Environment**
Peng Jiang, Cui Guo, Yonghui Dai
- 75 Machine Learning and Text Mining based Real-Time Semi-Autonomous Staff Assignment System**
ARSLAN, Yunus Emre IŞIK, Yasin GÖRMEZ, Mustafa TEMİZ
- 95 Activity Recognition for Elderly Care Using Genetic Search**
Ankita Biswal, Chhabi Rani Panigrahi, Anukampa Behera, Sarmistha Nanda, Tien-Hsiung Weng, Bibudhendu Pati, Chandan Malu
- 117 Comparing Semantic Graph Representations of Source Code: The Case of Automatic Feedback on Programming Assignments**
José Carlos Paiva, José Paulo Leal, Álvaro Figueira
- 143 MK-MSVCR: An Efficient Multiple Kernel Approach to Multi-class Classification**
Zijie Dong, Fen Chen, Yu Zhang
- 167 An Approach for Supporting Transparent ACID Transactions over Heterogeneous Data Stores in Microservice Architectures**
Lazar Nikolić, Vladimir Dimitrieski, Milan Ćeliković
- 203 SPC5: an efficient SpMV framework vectorized using ARM SVE and x86 AVX-512**
Evann Regnault, Bérenger Bramas

Special Section: Adaptive Smart Areas and Intelligent Agents

- 223 Evaluation of Deep Learning Techniques for Plant Disease Detection**
Cedric Marco-Detchart, Jaime Andrés Rincon, Carlos Carrascosa, Vicente Julian
- 245 A Flexible Approach for Demand-Responsive Public Transport in Rural Areas**
Pasqual Martí, Jaume Jordán, Vicente Julian
- 269 How to Fairly and Efficiently Assign Tasks in Individually Rational Agents' Coalitions? Models and Fairness Measures**
Marin Lujak, Alessio Salvatore, Alberto Fernández, Stefano Giordani, Kendal Cousy

Special Section: Role of Agents in Traffic and Transportation

- 291 Comparing Reinforcement Learning Algorithms for a Trip Building Task: a Multi-objective Approach Using Non-Local Information**
Henrique U. Gobbi, Guilherme Dytz dos Santos, Ana L. C. Bazzan
- 309 Sustainability-Oriented Route Generation for Ridesharing Services**
Mengya Liu, Vahid Yazdanpanah, Sebastian Stein, Enrico Gerding
- 335 Knowledge Transfer in Multi-Objective Multi-Agent Reinforcement Learning via Generalized Policy Improvement**
Vicente N. de Almeida, Lucas N. Alegre, Ana L. C. Bazzan

Special Section: Innovations in Intelligent Systems and Applications

- 363 3D Convolutional Long Short-Term Encoder-Decoder Network for Moving Object Segmentation**
Anil Turker, Ender Mete Eksioğlu
- 379 Echo State Network for Features Extraction and Segmentation of Tomography Images**
Petia Koprinkova-Hristova, Ivan Georgiev, Miryana Raykovska
- 395 VINIA: Voice-Enabled Intent-Based Networking for Industrial Automation**
Raul Barbosa, João Fonseca, Marco Araújo, Daniel Corujo

Editorial

Mirjana Ivanović, Miloš Radovanović, and Vladimir Kurbalija

University of Novi Sad, Faculty of Sciences
Novi Sad, Serbia
{mira,radacha,kurba}@dmi.uns.ac.rs

Kicking off the year 2024, this first issue of Volume 21 of Computer Science and Information Systems consists of 10 regular articles and three special sections: “Adaptive Smart Areas and Intelligent Agents” (3 articles), “Role of Agents in Traffic and Transportation” (3 articles) and “Innovations in Intelligent Systems and Applications” (3 articles). This editorial brings brief presentation of regular papers. Papers published in special sections are briefly presented in appropriate guest editorials.

As always, we are thankful for the hard work and enthusiasm of our authors, reviewers, and guest editors, without whom the current issue and the publication of the journal itself would not be possible.

In the first regular article, “FSASA: Sequential Recommendation Based on Fusing Session-Aware Models and Self-Attention Networks,” Shangzhi Guo et al. propose an approach for sequential recommendation based on Fusing Session-Aware models and Self-Attention networks (FSASA), where the Self-Attentive Sequential Recommendation (SASRec) model is used as a global representation learning module to capture long-term preferences under user behavior sequences.

The second regular article, “Applying SPIN Checker on 5G EAP-TLS Authentication Protocol Analysis” by Qianli Wang, applies the SPIN model checker to produce a formal analysis of the 5G EAP-TLS authentication protocol. The article provides a comprehensive understanding of the security properties of the 5G EAP-TLS protocol, and offers valuable insights and guidance for the verification of the protocol’s security properties, security design, and optimization of protocol implementation and interoperability.

“PI-BODE: Programmable Intraflow-based IoT Botnet Detection system,” by Đorđe D. Jovanović and Pavle V. Vuletić, proposes a programmable intraflow-based IoT botnet detection (PI-BODE) system based on the detection of Command and Control (C&C) communication between infected devices and the botmaster. Based on the analysis of the traffic intraflow statistical parameters, the approach allows detecting malicious communication before any attacks occur.

Peng Jiang et al., in “Feature Parameters extraction and Affective Computing of Voice Message for Social Media Environment” analyze the cognitive differences between semantic and acoustic features of voice messages from the perspective of cognitive neuroscience, and present a voice feature extraction method based on EEG (electroencephalogram) experiments, on top of which an affective computing method based on Pleasure-Arousal-Dominance (PAD) is proposed.

In “Machine Learning and Text Mining based Real-Time Semi-Autonomous Staff Assignment System,” Halil Arslan et al. present a machine learning-based decision support system for staff assignment that works with real-time data. The system analyses the description of newly requested tasks using text mining and machine-learning approaches,

predicts the optimal available staff that meets the needs of the project task, and iteratively updates personnel qualifications after each completed task.

The article “Activity Recognition for Elderly Care Using Genetic Search” by Ankita Biswal et al. proposes a model for human activity recognition (HAR) which provides a substructure for the assisted living environment, and uses genetic search based feature selection to manage the voluminous data generated from various embedded sensors. The article also presents a cloud based edge computing architecture for seamless deployment of the proposed model.

José Carlos Paiva et al., in “Comparing Semantic Graph Representations of Source Code: The Case of Automatic Feedback on Programming Assignments,” provide a thorough comparison of the most widespread semantic graph representations for the automated assessment of programming assignments, including usage examples, facets, and costs for each of these representations. A benchmark has been conducted to assess their cost using the Abstract Syntax Tree (AST) as a baseline. The results demonstrate that the Code Property Graph (CPG) is the most feature-rich representation, but also the largest and most space-consuming.

In “MK-MSVCR: An Efficient Multiple Kernel Approach to Multi-class Classification,” Zijie Dong et al. introduce a novel multi-class support vector classification and regression (MSVCR) algorithm with multiple kernel learning (MK-MSVCR) based on two-stage learning (MK-MSVCR-TSL). The two-stage learning aims to make classification algorithms better when dealing with complex data by using the first stage of learning to generate “representative” or “important” samples.

Lazar Nikolić et al., in “An Approach for Supporting Transparent ACID Transactions Over Heterogeneous Data Stores in Microservice Architectures,” present the Service Proxy Transaction Management (SPTM) approach to microservice architectures (MSA), which offers scalable reads and transactions with ACID (Atomicity, Consistency, Isolation, and Durability) guarantees. The novelty of the approach is in intercepting inbound messages to services, rather than having services directly communicate with a transaction manager.

Finally, “SPC5: An Efficient SpMV Framework Vectorized Using ARM SVE and x86 AVX-512” authored by Evann Regnault and Bérénger Bramas describes how the SPC5 sparse matrix/vector product (SpMV) framework was ported to the ARM-based AFX64 CPU by converting Intel AVX512 kernels to the Scalable Vector Extension (SVE) vectorization technology.

This first issue of the Journal brings a number of papers from diverse research domains and contemporary research fields. Accordingly we expect that wider audience will find at least one interesting paper for reading. Also we hope that papers will inspire readers to improve their research or even try to do innovative work in some of research areas presented in papers.

Guest Editorial: Adaptive Smart Areas and Intelligent Agents

Marin Lujak¹, Adriana Giret², and Sara Rodríguez González³

¹ CETINIA, University Rey Juan Carlos, Madrid, Spain
marin.lujak@urjc.es

² Universitat Politècnica de València, Valencia, Spain
agiret@dsic.upv.es

³ University of Salamanca, Salamanca, Spain
srg@usal.es

Intelligent agents showcase adaptability as computational entities that utilize sensors and actuators to navigate and manipulate different environments in pursuit of predefined objectives. One of the promising research fields on Agents and Multi-Agent Systems is in adaptive smart areas, which refer to spatial constructs with a high need for sensorization intentionally designed to be responsive and adaptable to changing environmental conditions and user needs and preferences. Unlike environments that merely tolerate external factors, the goal is to create an intelligent environment that actively responds to changes and seamlessly accommodates the evolving requirements and challenges, ensuring optimal functionality. The integration of intelligent agents within these adaptive environments amplifies their effectiveness, creating synergies that enhance the overall adaptability and responsiveness of both the computational entities and the designed systems. This interconnected approach underscores the importance of harmonizing the design of agents and multi-agent systems in the face of diverse and evolving conditions. Research in adaptive Smart Areas integrates cost efficiency, sustainable mobility, environmental protection, and economic sustainability including intelligent agents with unlimited opportunities to display their abilities to react, plan, learn, and interact in an intelligent and rational manner.

This Special Section contains a selection of revised and extended versions of the papers presented at the first Workshop on Adaptive Smart areaS and Intelligent Agents (ASSIA) held in conjunction with PAAMS 2022, the 20th International Conference on Practical Applications of Agents and Multi-Agent Systems in L'Aquila, Italy on the 14th July, 2022. The aim of the ASSIA 2022 workshop was to propose and discuss new agent technologies aimed at fostering collaboration and coordination and providing intelligence to Adaptive Smart Areas applied to urban and rural areas, buildings, farms, and forests, among others. The use of agents in Adaptive Smart Areas tackled issues related to smart architectures, simulations, intelligent infrastructure, smart transport, robotics, and open data. The workshop also addressed specific methodological and technological issues raised by the deployment of agents in the real-world Adaptive Smart Areas. This perspective offered a unique opportunity to advance the current state of the art, while addressing ongoing challenges in the field. The ASSIA 2022 workshop, as well as the PAAMS 2022 conference in which it took place, were held in person, with a very active and lively interaction of the presenters and the audience.

Following the standard reviewing procedure of the ComSIS Journal, three papers were accepted for publication in this section, and here is a brief overview of the papers included.

The first paper titled “Evaluation of Deep Learning Techniques for Plant Disease Detection” authored by C. Marco-Detchart, J. A. Rincon, C. Carrascosa and V. Julian, embarks on a comprehensive exploration of various deep learning techniques leveraging leaf images for the autonomous detection of pests and diseases in crops. By training with pictures of affected crops and healthy crops, deep learning techniques learn to distinguish one from the other. The authors utilize the Convolutional Neural Networks (CNN) including mobile-oriented network architectures (InceptionV3, MobileNetV2, and NasNetMobile) as well as high-performance computer-oriented CNNs (EfficientNet and Efficientnet-B0 and EfficientNetV2-B0). They consider the preprocessing step inspired by the Bezdek Breakdown Structure (BBS) for edge detection as the input to their neural system and use the Gravitational Smoothing (GS) process as a conditioning step in the preprocessing. By providing a deeper understanding of the strengths and limitations of these methodologies, this research contributes to the forefront of agricultural disease detection.

The second paper titled “A Flexible Approach for Demand-Responsive Public Transport in Rural Areas” by Pasqual Martí, Jaume Jordán and Vicente Julian studies on-demand mobility in rural areas characterized by low demand, long distance among settlements, and an older population. The authors propose a heuristic demand-responsive transportation system scheduler for offline and online allocation of travel requests to vehicles and a search algorithm for feasible insertions within available itineraries and test it in simulation. The simulation results highlight the clear potential of the demand-responsive mobility paradigm to serve rural demand at an acceptable quality of service and provide a modern, dynamic, and reliable means of public transportation to rural contexts.

Finally, the third paper titled “How to Fairly and Efficiently Assign Tasks in Individually Rational Agents’ Coalitions Models and Fairness Measures” titled by Marin Lujak, Alessio Salvatore, Alberto Fernández, Stefano Giordani and Kendal Cousy, studies coalitions of individually rational agents. Such coalitions can be observed in, e.g., agricultural cooperatives and taxi services. Since agents’ performance, costs, and skills may vary from task to task, the decisions about individual agent-task assignment will determine the overall performance of the coalition. They propose two new models that balance efficiency and fairness and study the utilitarian, egalitarian, and Nash social welfare for task assignment in this context. Moreover, they propose three new fairness measures based on equity and equality and use them to compare the newly proposed models. Through functional examples, the authors show that a reasonable trade-off between efficiency and fairness in task assignment is possible through the use of the proposed models.

Guest Editorial: Role of Agents in Traffic and Transportation

Marin Lujak¹, Ana L. C. Bazzan², Ivana Dusparic³, and Giuseppe Vizzari⁴

¹ CETINIA, University Rey Juan Carlos, Spain
marin.lujak@urjc.es

² Federal University of Rio Grande do Sul, Brasil
bazzan@inf.ufrgs.br

³ School of Computer Science and Statistics, Trinity College Dublin, Ireland
duspari@tcd.ie

⁴ Department of Informatics, Systems and Communication, University of Milano – Bicocca, Italy
giuseppe.vizzari@unimib.it

Intelligent algorithms, particularly those utilizing autonomous agents and multi-agent systems, offer a compelling solution to the pursuit of secure, efficient, and sustainable traffic and transportation. Even though many real-world problems are inherently distributed and multi-objective, most of the literature deals with a single agent and a single objective. Such approaches may generally present robustness issues, and difficulties in responding to the generally heterogeneous individual actors' conditions changing over time. Therefore, a need arises for a way to model and train multiple agents to tackle these challenges.

This special section explores synergies among agents, multi-agent systems, multi-objective optimization, and machine learning, offering valuable insights into combining their potential within the context of traffic and transportation. This unique perspective offers an opportunity to advance the current state of the art, addressing ongoing challenges in the field. The section introduces innovative agent-based methods designed for dynamic adaptation, ensuring the smooth and efficient operation of transportation systems, even under fluctuating conditions. In combination with [1], it contains a selection of revised and extended versions of papers presented at the 12th International Workshop on Agents in Traffic and Transportation (ATT 2022) held in conjunction with IJCAI-ECAI 2022, the 31st International Joint Conference on Artificial Intelligence and the 25th European Conference on Artificial Intelligence, in Vienna, Austria on July 25th 2022. The workshop aimed to unite researchers and practitioners to discuss the modeling, simulation, control, and management of large-scale transportation systems at both micro and macro levels. The ATT 2022 workshop, as well as the IJCAI-ECAI 2022 conference in which it took place, were held in person, with a very active and lively interaction of the presenters and the audience. ATT 2022 thus continued a long success story of the eleven previous workshop editions and the trend of publishing revised and extended selected papers in special issues [2].

Following the standard reviewing procedure of the ComSIS Journal, two papers have been accepted for publication in this section, with a third added due to topic similarity. In the following, we give a short overview of the papers contained in the special section.

The first paper entitled “Comparing Reinforcement Learning Algorithms for a Trip Building Task: a Multi-objective Approach Using Non-Local Information”, authored by Henrique U. Gobbi, Guilherme Dytz dos Santos and Ana L. C. Bazzan formulates the problem of multiple agents learning how to travel from an origin to a destination as a

reinforcement learning (RL) task modeled as a stochastic game. The authors consider more than one objective, non-stationarity, and inter-agent communication of local and non-local information. RL algorithms for a single objective (Q-learning) and multiple objectives (Pareto Q-learning) are compared, with and without non-local communication. These methods are evaluated with hundreds of agents, aiming at minimizing their origin-destination travel times and the carbon monoxide emissions. Results show that the use of non-local communication reduces both travel time and emissions.

The second paper titled “Sustainability-Oriented Route Generation for Ridesharing Services” authored by Mengya Liu, Vahid Yazdanpanah, Sebastian Stein and Enrico Gerding consider three pillars of sustainability: social, economic, and the environmental one. They present a multi-objective evolutionary approach based on the Non-dominated Sorting Genetic Algorithm for generating routing options in sustainable mobility on demand under six sustainable ridesharing objectives: travelling time, waiting time, overall/excess distance, travel cost, total emission, and working time balance. In addition to being aware of sustainability, their method also establishes a foundation for explainable, participatory, and dynamic mobility-on-demand services. Stakeholders can be provided with visualisations to see how different objectives affect routing solutions. A diverse range of solutions may be provided for a diverse set of users. Lastly, the proposed approach allows dynamic fine-tuning over time.

Finally, the third paper titled “Knowledge Transfer in Multi-Objective Multi-Agent Reinforcement Learning via Generalized Policy Improvement” by Vicente N. de Almeida, Lucas N. Alegre and Ana L. C. Bazzan proposes a multi-objective multi-agent reinforcement learning method in which agents build a shared set of policies during training, in a decentralized way, and then combine these policies using a generalization of policy improvement and policy evaluation to generate effective behaviors for any possible preference distribution, without requiring any additional training. This method is applied to two different application scenarios: a multi-agent extension of a four-room environment and traffic signal control considering both vehicles and pedestrians. Results show that the approach is able to effectively and efficiently generate behaviors for the agents, given any preference over the objectives.

References

1. Bazzan, A.L., Dusparic, I., Lujak, M., Vizzari, G.: Agents in Traffic and Transportation (ATT 2022): Revised and Extended Papers. *AI Communications (Preprint)*, 1–3 (2023), <https://doi.org/10.3233/AIC-239001>
2. Lujak, M., Dusparic, I., Klügl, F., Vizzari, G.: Agents in Traffic and Transportation (ATT 2020). *AI Communications* 34(1), 1–3 (2021), <https://doi.org/10.3233/AIC-201640>

Guest Editorial: Innovations in Intelligent Systems and Applications

Mirjana Ivanović¹, Richard Chbeir², and Yannis Manolopoulos³

¹ University of Novi Sad, Serbia

² Université de Pau et des Pays de l'Adour, France

³ Open University of Cyprus, Cyprus Dublin, Ireland

This special section includes extended versions of selected papers from the 15th International Conference on INnovations in Intelligent SysTems and Applications (INISTA), which has been held during August 8-10, 2022, in Biarritz, France in hybrid mode.

The series of INISTA conferences have been organized since 2005. INISTA Conferences aim to bring together researchers from the entire spectrum of the multi-disciplinary fields of intelligent systems and to establish effective means of communication between them. In particular, they focus on all aspects of intelligent systems and the related applications, from both perspectives: theory and practice. The topics of interest of INISTA 2022 covered the entire spectrum of the multi-disciplinary fields of intelligent systems and related applications. In particular, the topics included: Artificial Intelligence Algorithms, Artificial Neural Networks; Autonomous systems; Bioinformatics. Big Data Applications, Algorithms, and Systems; Cloud/Edge/Fog Computing; Computational and Data Science; Data Mining; Data Hiding; Deep Learning; Distributed Intelligence; Ensemble Learning; Evolutionary Computation; Expert Systems; Fuzzy Logic; Genetic Algorithms; Hardware Implementations for Intelligent Systems; Human-Computer Interaction; Humanoid Robotics; Hybrid Intelligence; Intelligent Agents; Intelligent Applications in Biomedical Engineering; Intelligent Approaches in Robotic and Automation; Intelligent Approaches in Signal and Image Processing; Intelligent Approaches in System Identification/Modeling; Intelligent Behavior; Intelligent Control Systems; Intelligent Defense/Security Systems; Intelligent Healthcare; Intelligent Education; Intelligent Interaction and Visualization; Intelligent Life; Information Security; Internet of Things, Internet of Everything; Machine Learning; Memetic Computing; Natural Language Processing; Neurotechnology and Emergent Intelligence in Nervous Systems; Robust Perception in Complex Environments; Reinforcement Learning; Smart Sensors, Materials, and Environments; Smart Wearables; Social Media Mining; Swarm Intelligence; Text Mining; Virtual, Augmented, and Mixed Reality; Other topics related to Intelligent Systems.

In 2022 year, there were 78 accepted papers in the conference, and 10 of them were selected and invited for this special section. Submitted papers were based on original conference papers. Moreover, they were carefully revised, extended, improved, and judged acceptable for publication in this journal. Each paper has undergone a review process of at least two rounds, as well as it has been reviewed by two or three referees. Finally, 3 papers were accepted for publishing. The aim of this special issue is to present some new directions and research results in the area of intelligent systems.

The first paper “3D Convolutional Long Short-Term Encoder-Decoder Network for Moving Object Segmentation” by Anil Turker and Ender M. Eksioğlu presented the MOS-Net (Moving Object Segmentation) deep framework, an encoder-decoder network that combines spatial and temporal features using the flux tensor algorithm, 3D CNNs, and

ConvLSTM in its different variants. In an enhanced version the framework MOS-Net 2.0, additional ConvLSTM modules are added to 3D CNNs for extracting long-term spatiotemporal features. In the final stage of the framework the output of the encoder-decoder network, the foreground probability map, is thresholded for producing a binary mask, where moving objects are in the foreground and the rest forms in the background. In addition, an ablation study has been conducted to evaluate different combinations as inputs to the proposed network, including challenging videos such as those with dynamic backgrounds, bad weather, and illumination changes. The results of the proposed approach are compared with other competitive methods from the literature using the same evaluation strategy, and it has been concluded that the introduced MOS networks give highly competitive results.

The second paper “Echo State Network for Features Extraction and Segmentation of Tomography Images” by Petia Koprinkova-Hristova, Ivan Georgiev and Miryana Raykovska proposed a novel approach for gray scale images segmentation. It is based on multiple features extraction from a single feature per image pixel, i.e., its intensity value, via a recurrent neural network (Echo state network). The preliminary tests on the benchmark gray scale image Lena demonstrated that the newly extracted features (i.e., reservoir equilibrium states) reveal hidden image characteristics. Additionally, the developed approach was applied to a real-life situation for segmentation of a 3D tomography image of a bone. The aim of this application was to explore the object’s internal structure. The achieved results confirmed that the novel approach allows for clearer revealing the details of the bone internal structure, thus, supporting further tomography image analyses. Obtained results are also valuable from the practical point of view.

The problem of integrating virtual assistants with Intent-based Networking (IBN) and Software-defined Networking (SDN) for industrial network automation is explored in the paper “Voice-Enabled Intent-Based Networking for Industrial Automation” by Raul Barbosa, Joao Fonseca, Marco Araujo and Daniel Corujo. This work presented a preliminary architecture for a voice-enabled IBN system. The proposed architecture included the support of network orchestrators and network slice managers in the existing solution to allow the configuration of more network assets, including 5G Networks, improving the system’s capabilities. The results presented in the paper provide insights into this solution’s potential benefits and limitations to enhance the automation of the management and orchestration procedures in industrial networks.

We gratefully acknowledge all the hard work and enthusiasm of authors and reviewers, without whom the special section would not have been possible. Also, we believe that readers will enjoy reading these papers and will be inspired for their future work.

FSASA: Sequential Recommendation Based on Fusing Session-Aware Models and Self-Attention Networks

Shangzhi Guo¹, Xiaofeng Liao¹, Fei Meng¹, Qing Zhao¹, Yuling Tang², Hui Li³ and Qinqin Zong^{4,*}

¹ College of Computer Science, Chongqing University
Chongqing 400044, China

{20211401018g, xfliao, 20211401024g, 20211401020g}@cqu.edu.cn

² Hunan Creator Information Technologies CO., LTD.
Changsha 410000, China
yuling.tang@chinacreator.com

³ Jiangxi Institute of Land and Space Survey and Planning
Nanchang 330000, China
lihcool@126.com

⁴ Jiangxi Biological Vocational College
Nanchang 330200, China
honeybabyqinqin@gmail.com

Abstract. The recommendation system can alleviate the problem of “information overload”, tap the potential value of data, push personalized information to users in need, and improve information utilization. Sequence recommendation has become a hot research direction because of its practicality and high precision. Deep Neural Networks (DNN) have the natural advantage of capturing comprehensive relations among different entities, thus almost occupying a dominant position in sequence recommendation in the past few years. However, as Deep Learning (DL)-based methods are widely used to model local preferences under user behavior sequences, the global preference modeling of users is often underestimated, and usually, only some simple and crude user latent representations are introduced. Therefore, this paper proposes a sequential recommendation based on Fusing Session-Aware models and Self-Attention networks (FSASA). Specifically, we use the Self-Attentive Sequential Recommendation (SASRec) model as a global representation learning module to capture long-term preferences under user behavior sequences and further propose an improved session-aware sequential recommendation model as a local learning representation module from user model the user’s dynamic preferences in the historical behavior, and finally use the Gated Recurrent Unit (GRU) module to calculate their weights. Experiments on three widely used recommendation datasets show that FSASA outperforms state-of-the-art baselines on two commonly used metrics.

Keywords: Recommendation Systems, Sequential Recommendation, Session-Aware Recommendation, Self-Attention, Gated Recurrent Unit.

* Corresponding author

1. Introduction

In recent years, with the rapid development of the Internet, especially the mobile Internet, Internet information has also shown an explosive growth trend. Faced with massive amounts of information, the time and cost for users to obtain the content they need have increased significantly. The recommendation system, as an effective means to solve the problem of information overload, has become the core of many e-commerce and multimedia platforms [60]. Personalized recommendation services can help the platform to attract users' attention, increase the number of user visits, and provide a steady stream of power for the development of network platforms. Its commercial value has also attracted the attention of industry and academia [19].

Collaborative Filtering (CF) is the most widely used recommendation system in the early stage. The core idea is to synthesize the explicit feedback information of users and items and filter out items the target users may be interested in for recommendation [42]. Different from CF, the goal of sequence recommendation is to combine a personalized model of user behavior (based on historical information) with some concept of "context" based on the user's recent behavior by understanding and analyzing the user's interaction history as a sequence information, to push the matching items of user interest [46, 52, 35]. Early work on sequential recommendation usually uses Markov Chain (MC) [8, 7, 10, 11], but its disadvantage is also obvious. Due to the Markov property, it is assumed that the current interaction only depends on one or a few recent interactions. Only short-term dependencies are captured, while long-term dependencies are ignored. As one of the important research directions of sequence recommendation, session-aware recommendation takes each session as the basic input unit, which can capture the user's short-term preference and the dynamic preference reflected by the interest transfer between sessions, thereby improving the accuracy and timeliness of recommendation [43, 49]. Deep neural network has the natural advantage of capturing the comprehensive relationship between different entities, which can alleviate the problem of insufficient expressive ability of traditional recommendation models, so it has almost occupied the dominant position in sequence recommendation in the past few years [28, 44, 61, 24, 25, 9, 4]. However, most of DNN-based methods also do not pay enough attention to the long-term relationship between sequences, and the user's global preference modeling is often underestimated. Attention mechanisms, which can reveal syntactic and semantic patterns between words in a sentence, have also become an important component of sequence recommendation [23, 2, 45, 30]. Among them, SASRec [18] stacks multiple self-attention blocks, which can effectively capture the long-term preferences of users within a sequence. Now that users' long-term and short-term preferences have been well explored in previous studies, an intuitive way to develop sequential recommendation methods is to model local dynamic preferences and combine them with global preferences to more comprehensively predict users' true preferences [23, 38, 27, 54, 59].

Inspired by the above work, this paper proposes a novel solution named sequential recommendation based on fusing session-aware models and self-attention networks (FSASA). FSASA can consider the user's long-term static preference and short-term dynamic preference simultaneously and more fully express the user's real intention. Specifically, our model contains three main components, a global representation learning module, a local representation learning module, and a GRU module. For global representation learning, we follow SASRec [18] based on the self-attention mechanism because it

achieves excellent performance in capturing users' long-term preferences. For global representation learning, we introduce implicit features from users' historical behaviors based on sequential recommendation [43] to model users' dynamic preferences accurately. Finally, we use GRU to balance the weights of the global representation learning module and the local representation learning module. In addition, we conduct a comprehensive ablation study to show the impact of crucial modules and parameters on recommendation performance.

The main contributions of the proposed FSASA are as follows:

- We propose a novel sequential recommendation based on session-aware models and self-attention networks to capture the dynamic preferences beneath users' behavior sequences, and improving recommendation performance.
- We design a session-aware local representation learning module for mining the implicit features in the user's historical behavior to model the user's dynamic preferences accurately.
- The GRU module is used to balance the contribution of the global representation learning module and the local representation learning module and more comprehensively predict the user's real preferences.
- To verify the performance of FSASA, we also conducted simulation experiments on three commonly used datasets. Experimental results show that FSASA significantly outperforms five state-of-the-art baselines. We also perform ablation studies and discuss details of local and gating units.

The rest of this paper is structured as follows. In the next section, a brief review is given of recent investigations on general recommendation, sequential recommendation, and session-aware recommendation. We propose sequential recommendation based on fusing session-aware models and self-attention networks in Section 3. Section 4 describes experiments based on three real datasets and analyzes the results. Finally, Section 5 presents the main conclusions and future work.

2. Related Work

In this section, we will briefly review several lines of works closely related to ours, including general recommendation, sequential recommendation, and session-aware recommendation, respectively, and point out the relationship and differences between our FSASA and those works.

2.1. General Recommendation

Collaborative filtering [42] was the most widely used recommendation algorithm in the early days. It mainly finds users' preferences through deep mining of their past behavioral data, groups users based on different preferences, and recommends items with similar tastes to other users in the group [20, 6, 51, 55]. Among them, the matrix decomposition [31, 21, 12, 1, 16, 62] algorithm uses Singular Value Decomposition (SVD), Eigenvalue Decomposition (ED), and other methods to decompose the co-occurrence matrix to generate an implicit vector for the user and the project, respectively, and uses the implicit

vector to represent the user’s interest and the project’s attributes, to dig the deep potential relationship between the user and the project—excellent performance in user rating prediction task. Collaborative filtering and matrix decomposition algorithms only utilize user-project interaction information. At the same time, Logistic Regression (LR) [39] can integrate user portrait features, item attributes, and context information, transform features into numerical vectors, input them into the network for training, learn the weight of each feature, and predict the probability of positive samples in the output layer. However, LR has limited characterization ability and does not carry out a cross combination of multiple features, which affects prediction accuracy. Rendle [37] proposed a Factorization Model (FM) by adding a second-order cross-feature combination based on logistic regression. The Facebook team [13] combined the gradient lifting decision tree with logistic regression and used the combined model to complete the recommendation task.

Combining deep learning and recommendation system can alleviate the problem of insufficient expression ability of the traditional recommendation model. Multi-Layer Perceptron (MLP) is a neural network with feed-forward structure. The data flows through the input layer and multiple hidden layers into the output layer to calculate the final result. The recommendation system often uses it to mine the crossover of high-order features and learn potential data patterns [40, 5, 3, 29].

2.2. Sequential Recommendation

Different from traditional collaborative filtering and content filtering-based recommendation systems, sequential recommendation attempts to model and understand user sequential behavior, interactions between users and items, and the evolution of user preferences and item popularity over time [50]. Early works on sequential recommendation usually use Markov chains, and MC’s natural advantage in modeling sequential dependencies provides an intuitive solution for sequential recommendation [8, 7, 10, 11]. Nevertheless, its shortcomings are also obvious. Due to the Markov property, it is assumed that the current interaction only depends on one or a few recent interactions, so it can only capture short-term dependencies and ignore long-term dependencies.

As mentioned in the previous section, deep neural networks have the natural advantage of capturing comprehensive relations among different entities (e.g., users, items, interactions). They thus have almost dominated sequential recommendation in the past few years. Recurrent Neural Network (RNN) is a deep network structure commonly used to process time series data. RNN can perform feed-forward calculation, maintain the information of the previous moment, and use historical state data and current state to predict output so that it can process sequence data such as text and audio [28, 44, 61, 24, 25]. To solve the problem of information loss caused by too long time intervals and the problem of gradient disappearance and explosion, RNN has also constructed new variants: Long Short-Term Memory network (LSTM) [9] and gated recurrent unit [4]. Our work uses GRU to fuse two representation learning modules, which will be discussed further in Section 3. RNN is not perfect, and it may only capture point dependencies and ignore set dependencies (e.g., several interactions collaborating to influence the next one). Since Convolutional Neural Networks (CNN) do not have strong sequential assumptions about the interactions in sequences, the above-mentioned shortcomings of RNN in sequence recommendation can be compensated to some extent. The CNN first puts all the embedding elements of the interaction into a matrix, then uses this matrix as an “image” in time and latent space,

and finally, learns sequential patterns as local features of the image, using convolutional filters for subsequent recommendations [47, 58, 57]. However, due to the limited size of filters used in CNN, CNN-based sequence recommendations cannot effectively capture long-term dependencies, which limits their applications. With the rapid development of Graph Neural Networks (GNN), numerous researchers utilize GNN to model and capture complex transition sequences of user-item interactions [26, 34, 52, 53]. This method fully exploits the advantages of GNN to capture complex relations in structured relational datasets, showing great potential for explainable recommendation, which is still in the early exploration stage.

The attention model is also often used in sequence recommendation to emphasize those genuinely relevant and essential interactions in the sequence while ignoring those interactions that are irrelevant to the next interaction, allowing the model to focus on more important information, reducing the impact of data noise on the impact of the results [18, 23, 2, 45, 30]. In this paper, we base the global representation learning module on the Self-Attention Sequential Recommendation (SASRec) model [18], which is an excellent sequential recommendation model. Note that [23] bases the local representation learning module on the SASRec model, which is similar to our FSASA and will be further discussed in Section 4.

2.3. Session-aware Recommendation

Sequential recommendation considers that all historical interaction information is equally important for predicting user's current preference. However, user preference may change over time, which is dynamic rather than static. Therefore, Session-Based Recommendation Systems (SBRS) have been proposed to bridge these gaps in recent years. SBRS takes each session as the basic input unit, which can capture the user's short-term preferences and the dynamic preferences reflected by the interest transfer between sessions, thereby improving the accuracy and timeliness of recommendations [49]. Unlike session-based, the session-aware recommendation is a method that uses the relationship between sessions for each user and makes recommendations by structurally decoupling long-term and short-term preferences from a slightly more diverse perspective [22]. [17] propose an early SBRS emphasizing the importance of considering recently observed user behavior when making recommendations. [36] proposed one of the earliest deep learning techniques for the session-aware recommendation, where the authors used two parallel GRU layers to model information across sessions. In the same year, Ruocco et al. [41] proposed the IIRNN model, which, like [36], uses the RNN architecture and extends session-based techniques to model inter-session and intra-session notifications. RNN were later also used in NSAR models [33] to encode session patterns combined with user embeddings to represent long-term user preferences across different sessions. Hu et al. [15] combine inter-session and intra-session context with a joint context encoder for item prediction. In [56], the authors utilize a two-layer hierarchical attention network to model short-term and long-term user interests. In [14], the authors are inspired by language modeling methods such as word2vec to treat items as words and recommend related items based on contextual information.

As the session-aware method is widely used in local and dynamic preference modeling under user behavior sequences, the user's global and static preference modeling is often underestimated. Usually, only some simple and crude user potential representations

are introduced. In our work, the respective advantages of session-aware and self-attention-based sequential recommendation methods are fully combined to model the user’s short-term and long-term preferences, respectively. This means that our FSASA can more comprehensively represent the user’s true intent, which we discuss further in the next section.

3. Proposed Method: FSASA

This section proposes our FSASA, i.e., sequential recommendation based on fusing session-aware models and self-attention networks. For sequential recommendation, we are given a user’s action sequence $\mathcal{S}^u = \{\mathcal{S}_1^u, \mathcal{S}_2^u, \dots, \mathcal{S}_t^u, \dots, \mathcal{S}_{|\mathcal{S}^u|}^u\}$, $u \in \mathcal{U}$, $\mathcal{S}_t^u \in \mathcal{I}$, where \mathcal{U} denote a set of users and \mathcal{I} denote a set of items. Given the interaction history \mathcal{S}_t^u , sequential recommendation aims to predict the item that user u will interact with at time step \mathcal{S}_{t+1}^u . In this paper, we use capital letters in bold to denote matrices and their lowercase form to denote the corresponding row vectors.

3.1. Global Representation Learning

First of all, we fix the input sequence of each user u by extracting his/her latest n behaviors, which is abbreviated as $\mathcal{S}^u = \{s_1, s_2, \dots, s_n\}$, where n represents the maximum length that can handle. If the sequence length exceeds n , we consider the most recent n actions. If the sequence length is less than n , we repeatedly add a padding item $\mathbf{0}$ on the left until the length is n . Let $\mathbf{M} \in \mathbb{R}^{|\mathcal{I}| \times d}$ denote the learnable item embedding matrix with d as the latent dimensionality. We can then represent the input sequence as an embedding matrix $\mathbf{E} \in \mathbb{R}^{n \times d}$, where $\mathbf{E}_i = \mathbf{M}_{s_i}$. A constant zero vector $\mathbf{0}$ is used as the embedding for the padding item.

Following [18], since the self-attention model does not include any recurrent or convolutional module, it is unaware of the positions of previous items. Hence we inject a learnable position embedding matrix $\mathbf{P} = [p_1; p_2; \dots; p_n] \in \mathbb{R}^{n \times d}$ to the input embedding matrix $\mathbf{E} \in \mathbb{R}^{n \times d}$, and obtain an input matrix $\mathbf{X}^{(0)} = [x_1; x_2; \dots; x_n] \in \mathbb{R}^{n \times d}$ for the self-attention network:

$$\mathbf{x}_i^{(0)} = \mathbf{m}_{s_i} + \mathbf{p}_i, i \in \{1, 2, \dots, n\} \quad (1)$$

Then, we feed the sequence $\mathbf{X}^{(0)} \in \mathbb{R}^{n \times d}$ into a series of stacked self-attention blocks (SABs). The output of the b -th block is as follows:

$$\mathbf{X}^{(b)} = SAB^{(b)}\mathbf{X}^{(b-1)}, b \in \{1, 2, \dots, B\} \quad (2)$$

Omitting the normalization layers with residual connection, each self-attention block can be viewed as a self-attention layer $SAL(\cdot)$ followed by a feed-forward layer $FFL(\cdot)$ as follows:

$$SAB(\mathbf{X}) = FFL(SAL(\mathbf{X})) \quad (3)$$

$$\mathbf{X}' = SAL(\mathbf{X}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\Delta \cdot \mathbf{V} \quad (4)$$

$$FFL(\mathbf{X}') = \text{ReLU}(\mathbf{X}'\mathbf{W}_1 + \mathbf{1}^T\mathbf{b}_1)\mathbf{W}_2 + \mathbf{1}^T\mathbf{b}_2 \quad (5)$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ is the position-aware input matrix, $\mathbf{Q} = \mathbf{X}\mathbf{W}_Q$, $\mathbf{K} = \mathbf{X}\mathbf{W}_K$ and $\mathbf{V} = \mathbf{X}\mathbf{W}_V$ with $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d \times d}$ are the projected query, key and value matrices, respectively, to improve the flexibility. Note that $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d \times d}$ and $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^{1 \times d}$ are weights and biases for the two layers of convolution, $\mathbf{1}$ is a unit row vector of size $1 \times n$ and Δ is the causality mask (i.e., a unit lower triangular matrix of size $n \times n$), to preserve the transitions from previous steps only.

[23] uses a simple location-based attention mechanism as a global representation learning module to model the user’s long-term static preferences. Its performance is not as good as that of SASRec, that stacks multiple self-attention blocks. It is shown in [18] that hierarchy is important for global representation. Specifically, self-attention blocks at the bottom tend to capture long-term dependencies, while higher blocks may focus on more recent dependencies. In this module, we use the bottom self-attention block $SAB^{(1)}(\cdot)$ as the global representation learning module of FSASA.

3.2. Local Representation Learning

While the self-attention block at the top of [18] can also be used to model a user’s short-term dynamic preferences, in many online services, user interactions are often grouped by sessions where preferences are likely to be shared, and this is where session-aware is needed to establish connections for each user’s session. Inspired by [46], we added user rating information embedding based on [43] to accurately model short-term dynamic preferences of users. Note that the local representation learning module is independent of the global one.

The simplest way to distinguish sessions within item sequences is to insert a separator between item sequences [27]. A learnable extra token called Session Token (ST) is inserted between sessions as if it were an item embedding. Unlike the fill marker, it is not excluded from attention, and it has the effect of moving the embedding one position per session. The advantage of this method is that it can indicate at inference time whether the input is a new session or not.

User rating information is one of the important criteria for modeling user preferences, but most current recommendation algorithms define rating as positive feedback, which is unreasonable. For a user, scoring an item only means that the user has browsed the item rather than that the user is interested in the item. For example, suppose a user gives an item a shallow score. In that case, the user is not interested in the item, and the recommendation system should recommend fewer such items.

In FSASA, the rating information is fine-grained, and the learnable Rating Segment Embedding (RSE) is used, representing the session’s importance and providing a sequence hierarchy. Note that, similar to session tokens, scoring information can also indicate whether it is a new session or not at inference time. For p -th item i in j -th session of a user, our input representation becomes: $x = IE_i + PE_p + RSE_j$, where IE is an item embedding, PE is a positional embedding from BERT, and RSE is a session segment embedding. The maximum number of sessions is limited so that only the most recent m sessions are considered. As in the implementation of positional embedding, ordinals are attached in the most recent order and padding is filled to match the model input length L .

For a timestamp t , we define a Temporal Encoding (TE) as follows:

$$TE(t) = [\cos(\omega_1 t + \theta_1) \cdots \cos(\omega_{d_T} t + \theta_{d_T})]^\top \quad (6)$$

where d_T is a temporal dimension, and ω_i, θ_i are learnable parameters. We concatenate temporal encoding vectors \mathbf{t} to the input representation \mathbf{X} , which gives us a Temporal Self-Attention (TAS) as follows:

$$TSA(\mathbf{X}, \mathbf{t}) = \text{softmax}\left(\frac{[\mathbf{X}\mathbf{t}][\mathbf{X}\mathbf{t}]^\top}{\sqrt{d_{\mathbf{X}} + d_{\mathbf{t}}}}\right)\mathbf{X} \quad (7)$$

where $d_{\mathbf{X}}$ is an input dimension of \mathbf{X} . Here we can see that the attention weight a_{ij} between (x_i, t_i) and (x_j, t_j) is calculated as:

$$a_{ij} = x_i^\top x_j + TE(t_i)^\top TE(t_j) \quad (8)$$

The weight becomes sum of self-attentiveness and temporal attentiveness [54]. For multi-layered and multi-headed Transformer layers, we concatenate TE on each layer and head. Note that TE can be trained on each layer or head separately, but empirically no significant improvements were found.

[23] uses SASRec as a local representation learning module to model users' short-term preferences, while SASRec is better at modeling recent activities of sparse datasets, and it is difficult to model normal or dense Recent activity on the dataset accurately. FSASA uses BERT-STR as a local representation learning module and adds user rating information. Thanks to the excellent representation ability of session-aware, it can accurately model the short-term dynamic preferences of users.

The input representation layer including all proposed methods is shown in Figure 1. The rest part of the model is identical to BERT4Rec [46]. Note that the difference from SASRec [18], which uses an autoregressive decoder, is that information other than item embedding such as positional embedding, session segment embedding, and temporal encoding can be utilized at inference time for the to-be-predicted item.

3.3. Gating Unit

To combine the local representation and the global representation, we may naturally think of concatenation or summation. Many researchers suggest a weighted summation to balance the two representations by considering the consistency of the item lists (corresponding to the sequences in our case), which performs better in their cases. Inspired by [23, 4], we use GRU to combine the weights of global and local representation learning module, and the fusion equation is as follows:

$$x(t) = x_{global} \otimes r + x_{local} \otimes (1 - r) \quad (9)$$

GRU is a variant of traditional RNN. Like LSTM, it can effectively capture the semantic association between long sequences and alleviate the phenomenon of gradient disappearance or explosion. At the same time, its structure and calculation are simpler than LSTM. Its core structure is composed of update gate $z(t)$ and reset gate $r(t)$:

$$z(t) = \sigma(\mathbf{W}_z \cdot [h_{t-1}, x_t]) \quad (10)$$

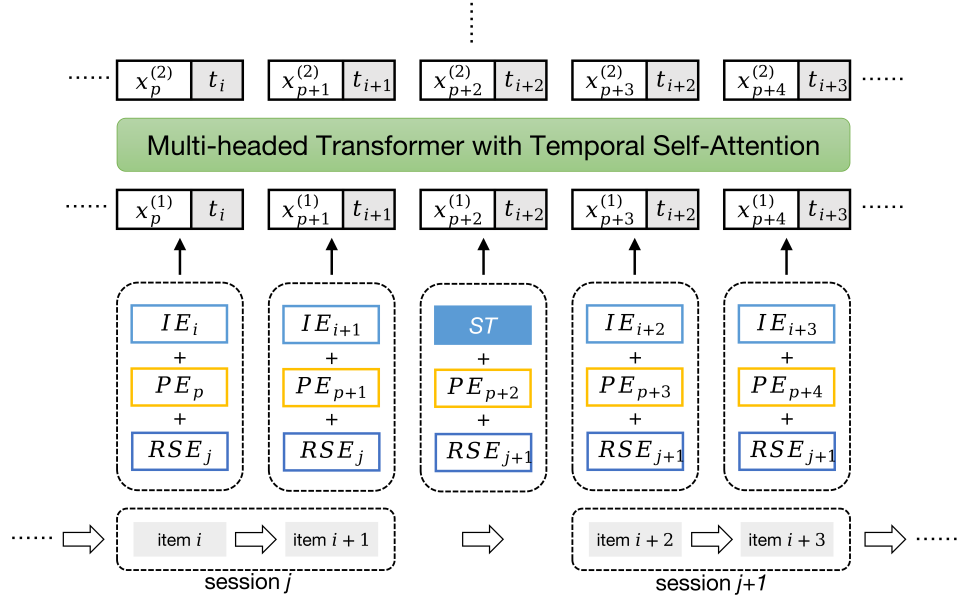


Fig. 1. Input layer

$$r(t) = \sigma(\mathbf{W}_r \cdot [h_{t-1}, x_t]) \quad (11)$$

where h_{t-1} denote the hidden state layer output of the previous time, in FSASA represents the user preference learned at the previous time, and x_t denote the input at the current time, which in FSASA represents the current local/short-term preference. After getting the gating signal, first use the reset gating to get the data after “RESET”:

$$h'_{t-1} = h_{t-1} \odot r_t \quad (12)$$

The representative controls how much information from the last time can be used. Then use this reset h'_{t-1} to perform basic RNN calculations, i.e., splicing with x_t for linear change, and after \tanh activation obtain h'_t :

$$h'_t = \tanh(\mathbf{W} \cdot [h'_{t-1}, x_t]) \quad (13)$$

The gate value z_t of the last update gate will act on the h'_t , and $1 - z_t$ will act on h_{t-1} , and then add the results of the two to get the final hidden state output h_t :

$$h_t = z_t \odot h'_t + (1 - z_t) \odot h_{t-1} \quad (14)$$

The range of the gating signal (i.e., z_t) is 0 to 1, the closer the gating signal is to 1, the more data is “remember”, and the closer to 0 is the more “forget”. FSASA uses GRU to “forget” the unimportant information in the user sequence, “remember” the important information in the user sequence, and more comprehensively and accurately represent the user’s true intention.

We abandon the way of compressing the long-term or short-term preference representation as the initial hidden state h_0 , as this would compress the representation vector to very low dimensions and lose information.

[23] uses item similarity models to add user’s uncertain intention information to make recommendations, although it can to a certain extent, the recommendation performance in the absence of user sequence information is improved. However, at the same time, noise may be introduced to affect the overall recommendation performance. FSASA directly uses GRU to balance the weight of the global representation module and the local representation module. Although the improvement of the recommendation performance is limited, it will certainly not have a negative impact on the recommendation performance and has robust scalability.

4. Performance Evaluation

To verify the performance of our proposed FSASA, this section introduces the details of the datasets, evaluation indicators, baseline methods, and parameter settings used in the simulation experiments. It conducts many ablation experiments to explore the impact of relevant hyperparameters on the performance of FSASA.

4.1. Datasets Description and Preprocessing

- **Steam**⁵[18, 48, 32]: This dataset contains data from October 2010 to January 2018 of Steam, a large online video game distribution platform. These include 2,567,538 users, 15,474 games, and 7,793,069 user reviews. The dataset also provides rich hidden information such as user’s game time, price information, purchase information, media ratings, categories, product bundles, developers, etc.
- **ML-1M**⁶: The dataset contains 1 million ratings of 4,000 movies from 6,000 users. This data includes movie ratings, movie metadata (genre and year), and user demographic data (age, zip code, gender, occupation, etc.).
- **ML-20M**⁷: The dataset contains 20,000,263 ratings and 465,564 tags for 27,278 movies from 138,493 users. Users are randomly selected, and each selected user has rated at least 20 movies. There is no demographic information, and each user is only given an ID, and no other private information is involved.

For sequential recommendation, we preprocess these datasets as follows:

1) To improve the dataset’s quality, we delete items with less than 5 interactions and delete users with less than 5 interactions; 2) When users have no new interactions within a day, use unix timestamp units to divide sessions. Each user has at least 2 sessions, each session contains at least 2 items, and only uses 200 recently interacted items; 3) In the preprocessing step, each comment or rating information is considered as a There is a hidden positive interaction, so this paper retains the rating in the data set when constructing the training set, which is an important improvement of this paper. For each user, the last

⁵ https://cseweb.ucsd.edu/~jmcauley/datasets.html#steam_data

⁶ <https://grouplens.org/datasets/movielens/1m/>

⁷ <https://grouplens.org/datasets/movielens/>

item is used as the test item, the second closest item is used as the validation, and the remaining items are used as the training set.

The statistics of the processed datasets are summarized in Table 1.

Table 1. Dataset statistics after preprocessing

Dataset	#Users	#Items	#Interactions	Avg. Length	Density
Steam	6330	4331	49163	7.77	0.18%
ML-1M	1196	3327	158496	132.52	3.98%
ML-20M	23404	12239	1981866	84.68	0.69%

4.2. Evaluation Metrics

We evaluate the recommendation performance via two standard metrics, i.e., recall (Recall@10, R@10) and normalized discounted cumulative gain (NDCG@10, N@10). Recall is how much of the information the user interested is predicted. The NDCG is a standardized DCG that considers the list of recommendations and the number of truly valid results in each search. The definition of Recall@10 and NDCG@10 are as follows:

$$Recall@10 = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \quad (15)$$

$$NDCG@10 = \frac{DCG@10}{IDCG@10} \quad (16)$$

$$DCG@10 = \sum_{i=1}^{10} \frac{2^{rel_i} - 1}{\log_2(i + 1)} \quad (17)$$

Where $R(u)$ represents the $Top-10$ recommendation list made to the user according to the user’s behavior in the training set, and $T(u)$ represents the item set actually selected by the user after the system recommends the item to the user. rel_i stands for correlation degree of items in position i , $IDCG@10$ stands for ideal DCG, i.e., DCG under perfect result.

4.3. Baselines

To verify the effectiveness of FSASA, we compare it with the following five representative baselines:

- **SASRec[18]**: It addresses the sequential recommendation problem by introducing a self-attention mechanism that adaptively assigns weights to previous entries at each time, tending to consider long-term dependencies on dense datasets while focusing on recent activities on sparse datasets. It is also a global representation learning module in our FSASA.

- **B4Rec[46]**: It employs deep bidirectional self-attention to model user behavior sequences. Each layer utilizes all the information of the previous layer and can capture the information of the entire field.
- **BERT-ST[43]**: It proposes three ways to utilize session information in the BERT-based model to improve sequential recommendation performance. We use the method with the best comprehensive performance among them to compare with FSASA.
- **BERT-STR**: The rating information is added based on BERT-ST, which is also the local representation learning module in our FSASA.
- **FISSA[23]**: From the global and local time series perspective, SASRec is used as a local learning module, and then a position-based attention layer is used as a global module, and their weights are balanced by gating.

4.4. Implementation Details

We perform all the experiments on a single server with ADM Ryzen5 3600x CPU and Nvidia 3070 GPU. The software environment includes Cuda 11.4, Cudnn 8.2, Miniconda 3, Python 3.7, deep learning framework Pytorch 1.10, and tensorboardx 2.5. All hyperparameters were tuned through grid search, and we report the one with the best performance in the final result.

The FSASA model comprised [18] and improved [43], respectively, to act as a global representation learning module and a local representation learning module. The former is used to capture long-term dependencies between items, while the latter is used to obtain short-term associations between items, using GRUs to balance their weight. We use the AdamW optimizer to calculate and update the model parameters to minimize the objective function, the learning rate is initialized to 0.001, and the dropout is set to 0.2 to avoid model overfitting. Limited by hardware conditions, the layer of SASRec is uniformly set to 1, and the parameter settings for different datasets are shown in Table 2.

Table 2. Initialization parameters for the three datasets

Dataset	MaxLength	Layers	Hidden_dim	Heads	Batch_size
Steam	15	2	128	2	1024
ML-1M	200	2	256	2	128
ML-20M	100	4	256	4	128

4.5. Overall Performance Comparison

Table 3 presents the recommendation performance of all methods on the three datasets. As we can see, here following observations would be found:

Compared with all baselines, our FSASA achieves the best performance on all three datasets, which clearly demonstrates the superiority of FSASA (note that the Steam dataset does not provide user rating information, so we use GRU to fuse the original SASRec and BERT-ST, but its performance still has a noticeable improvement). The second best performance is obtained by BERT-ST or BERT-STR, which is consistent with the observations of previous studies [46, 43, 23], which also show the advantages of session-aware

in modeling user dynamic preferences. In addition, BERT-STR performs slightly better than BERT-ST, illustrating that adding user rating information helps to accurately model truly relevant and important interactions in user sequences.

It is worth noting that FISSA uses a structure similar to FSASA, but its recommendation performance is not outstanding, only better than SASRec and basic B4Rec. We analyze the reasons from the components of FISSA and FSASA: 1) FISSA uses SASRec as a local representation learning module to model users' short-term preferences, while SASRec is better at modeling recent activities of sparse datasets, and it is difficult to model normal or dense Recent activity on the dataset accurately. FSASA uses BERT-STR as a local representation learning module and adds user rating information. Thanks to the excellent representation ability of session-aware, it can accurately model the short-term dynamic preferences of users; 2) FISSA uses a simple location-based attention mechanism as a global representation learning module to model the user's long-term static preferences. Its performance is not as good as that of SASRec, that stacks multiple self-attention blocks; 3) FISSA uses item similarity models to add user's uncertain intention information to make recommendations, although it can to a certain extent, the recommendation performance in the absence of user sequence information is improved. However, at the same time, noise may be introduced to affect the overall recommendation performance. FSASA directly uses GRU to balance the weight of the global representation module and the local representation module. Although the improvement of the recommendation performance is limited, it will certainly not have a negative impact on the recommendation performance and has robust scalability.

4.6. Ablation Study

We discuss the impact of relevant parameters on the performance of FSASA in this section.

(1) Effect of representation learning ratio

To explore the impact of the representation learning module in FSASA on recommendation performance, we manually set the proportion of the global representation learning module involved in FSASA. As shown in Figure 2, on the whole, with the increase of the proportion of the global representation learning module involved, the performance of FSASA fluctuates slightly before 0.5, and the performance decreases with the increase of the proportion after 0.5. Between 0.3 to 0.5, the performance of FSASA is optimal. It shows that the session-aware-based local representation learning module we proposed is dominant in FSASA. However, it also needs the assistance of the global representation learning module to more fully represent the user's true intentions.

(2) Effect of gating unit

To explore the impact of the gating unit in FSASA on the recommendation performance, we fixed the proportion of the global representation learning module and the local representation learning module at 0.5 for comparative experiments. As shown in Figure 3, the two metrics of FSASA using GRU outperform FSASA without GRU on all three datasets. The analysis in 4.6.1 shows that although the local representation learning module is dominant in FSASA, the larger the proportion, the better. GRU needs to be dynamically adjusted according to different scenarios to give full play to the greatest advantages of FSASA.

Table 3. Recommendation performance of FSASA and five baselines on three datasets. The best performing method in each row is bolded, and the second best performing method in each row is underlined.

Dataset	Metric	Item	SASRec	B4Rec	BERT-ST	BERT-STR	FISSA	FSASA
Steam ¹	R@10	Ran ²	0.7834	0.7987	<u>0.8120</u>	\	0.8015	0.8139
		Pop ³	0.5523	0.5670	<u>0.6030</u>	\	0.5735	0.6056
		All ⁴	0.5164	0.5313	<u>0.5616</u>	\	0.5425	0.5718
	N@10	Ran	0.6726	0.6915	<u>0.7093</u>	\	0.7002	0.7165
		Pop	0.5007	0.5196	<u>0.5596</u>	\	0.5316	0.5666
		All	0.4610	0.4782	<u>0.5187</u>	\	0.4885	0.5282
ML-1M	R@10	Ran	0.7199	0.7341	0.7291	<u>0.7400</u>	0.7380	0.7558
		Pop	0.4189	0.4725	0.4841	<u>0.4849</u>	0.4731	0.5192
		All	0.1480	0.1129	<u>0.1731</u>	0.1697	0.1528	0.1811
	N@10	Ran	0.4962	0.5100	0.5113	<u>0.5115</u>	0.5112	0.5177
		Pop	0.2674	0.3011	<u>0.3105</u>	0.3093	0.3021	0.3417
		All	0.0742	0.0508	0.0838	<u>0.0873</u>	0.0806	0.0915
ML-20M	R@10	Ran	0.9014	0.9053	<u>0.9114</u>	0.9113	0.9083	0.9168
		Pop	0.4370	0.4729	0.4799	<u>0.4822</u>	0.4735	0.5195
		All	0.1389	0.1381	0.1439	0.1393	<u>0.1499</u>	0.1555
	N@10	Ran	0.6954	0.6944	0.6910	0.6964	<u>0.6998</u>	0.7130
		Pop	0.2839	0.3051	0.3125	<u>0.3155</u>	0.3062	0.3588
		All	0.0707	0.0724	0.0754	0.0755	<u>0.0785</u>	0.0846

¹ Note that the Steam dataset does not provide user ratings.

² Randomly select 100 non-repeated items from the items that the user unclicked for recommendation.

³ Sort the item list in descending order, and continuously extract 100 unclicked and non-repeated items for recommendation.

⁴ The recommended label space is all items.

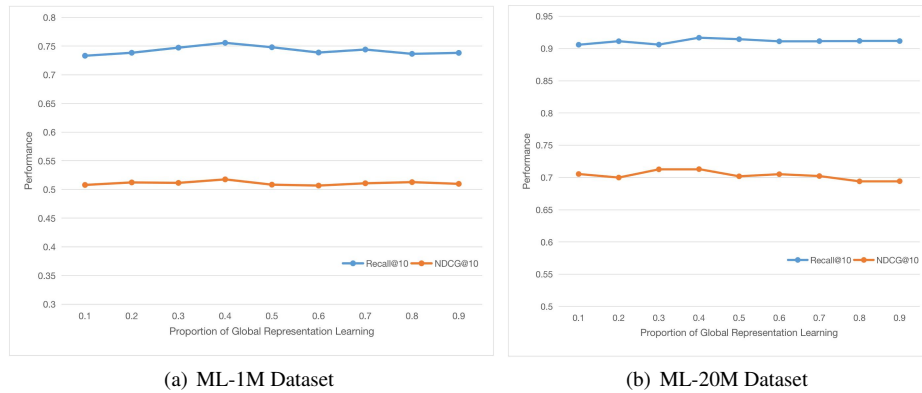


Fig. 2. Effect of representation learning ratio

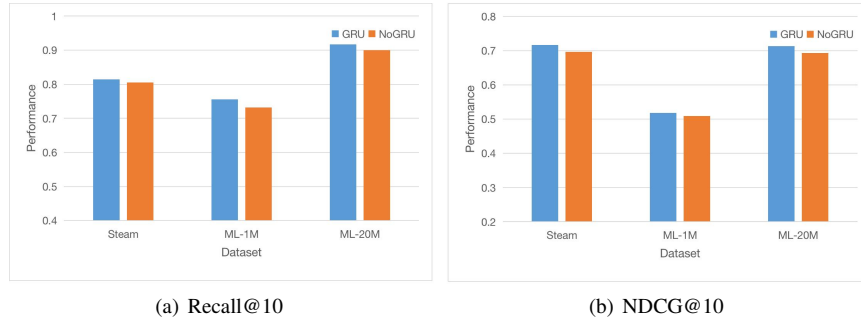


Fig. 3. Effect of gating unit

(3) Effect of rating

We add user rating information based on [43] to serve as the local representation learning module of FSASA. In order to verify the rationality of our work, we compare it with FSASA using the BERT-ST module. As shown in Figure 4, both metrics of FSASA using BERT-STR slightly outperform FSASA using BERT-ST on both datasets. Because BERT-STR adds the implicit feature of user rating information, it can more accurately model the user's short-term dynamic preferences.

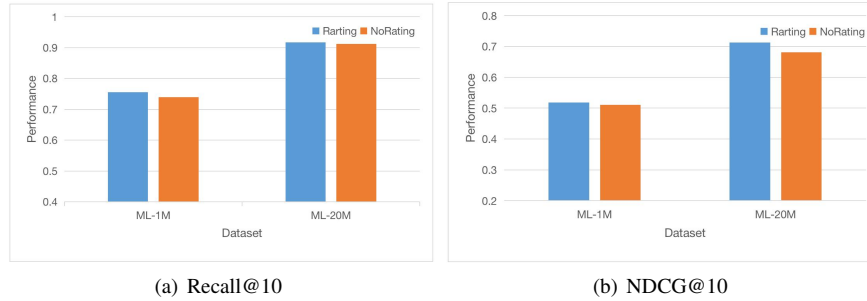


Fig. 4. Effect of rating

5. Conclusions And Future Work

With the rapid development of information technology, recommendation system plays an important role in alleviating the problem of information overload. Compared with traditional recommendation algorithms, deep learning enhances the model's scalability and representation ability, allowing the model to incorporate more diverse features. However, users' global preferences and modeling are often underestimated. Therefore, in this paper, we propose a new sequential recommendation method (FSASA) based on the session-aware model and self-attention network to capture global preferences and dynamic pref-

erences under user behavior sequences. Precisely, our model consists of three main components, the local presentation learning module, the global presentation learning module, and the GRUs module. We used the SASRec model as a global presentation learning module to capture long-term preferences under user behavior sequences and proposed an improved session-aware sequential recommendation model as a local learning presentation module to model users' dynamic preferences from users' historical behaviors. Finally, the Gated Recurrent Unit module is used to balance the weights of the two modules. We compared the FSASA model with various mainstream algorithms on three publicly available data sets. The results showed that the FSASA model was superior to the existing mainstream recommended model. We also conducted many ablation experiments and quantitative studies to demonstrate the rationality of the FSASA model. In future work, we plan to extend the model by incorporating rich contextual information, exploiting session information more thoroughly and consistently to predict users' future preferences accurately.

Acknowledgments. This work is supported by the Postgraduate Scientific Research and Innovation Foundation of Chongqing under Grant No. CYB22064.

References

1. Chen, C., Zhang, M., Zhang, Y., Liu, Y., Ma, S.: Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems (TOIS)* 38(2), 1–28 (2020)
2. Chen, J., Zhang, H., He, X., Nie, L., Liu, W., Chua, T.S.: Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In: *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. pp. 335–344 (2017)
3. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhya, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al.: Wide & deep learning for recommender systems. In: *Proceedings of the 1st workshop on deep learning for recommender systems*. pp. 7–10 (2016)
4. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014)
5. Covington, P., Adams, J., Sargin, E.: Deep neural networks for youtube recommendations. In: *Proceedings of the 10th ACM conference on recommender systems*. pp. 191–198 (2016)
6. Cui, Z., Xu, X., Fei, X., Cai, X., Cao, Y., Zhang, W., Chen, J.: Personalized recommendation system based on collaborative filtering for iot scenarios. *IEEE Transactions on Services Computing* 13(4), 685–695 (2020)
7. Feng, S., Li, X., Zeng, Y., Cong, G., Chee, Y.M.: Personalized ranking metric embedding for next new poi recommendation. In: *IJCAI'15 Proceedings of the 24th International Conference on Artificial Intelligence*. pp. 2069–2075. ACM (2015)
8. Garcin, F., Dimitrakakis, C., Faltings, B.: Personalized news recommendation with context trees. In: *Proceedings of the 7th ACM Conference on Recommender Systems*. pp. 105–112 (2013)
9. Graves, A., Graves, A.: Long short-term memory. *Supervised sequence labelling with recurrent neural networks* pp. 37–45 (2012)
10. He, R., Kang, W.C., McAuley, J.: Translation-based recommendation. In: *Proceedings of the eleventh ACM conference on recommender systems*. pp. 161–169 (2017)

11. He, R., McAuley, J.: Fusing similarity models with markov chains for sparse sequential recommendation. In: 2016 IEEE 16th international conference on data mining (ICDM). pp. 191–200. IEEE (2016)
12. He, X., Zhang, H., Kan, M.Y., Chua, T.S.: Fast matrix factorization for online recommendation with implicit feedback. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. pp. 549–558 (2016)
13. He, X., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., Shi, Y., Atallah, A., Herbrich, R., Bowers, S., et al.: Practical lessons from predicting clicks on ads at facebook. In: Proceedings of the eighth international workshop on data mining for online advertising. pp. 1–9 (2014)
14. Hu, L., Cao, L., Wang, S., Xu, G., Cao, J., Gu, Z.: Diversifying personalized recommendation with user-session context. In: IJCAI. pp. 1858–1864 (2017)
15. Hu, L., Chen, Q., Zhao, H., Jian, S., Cao, L., Cao, J.: Neural cross-session filtering: Next-item prediction under intra-and inter-session context. *IEEE Intelligent Systems* 33(6), 57–67 (2018)
16. Hwang, W.S., Li, S., Kim, S.W., Lee, K.: Data imputation using a trust network for recommendation via matrix factorization. *Computer Science and Information Systems* 15(2), 347–368 (2018)
17. Jannach, D., Ludewig, M., Lerche, L.: Session-based item recommendation in e-commerce: on short-term intents, reminders, trends and discounts. *User Modeling and User-Adapted Interaction* 27, 351–392 (2017)
18. Kang, W.C., McAuley, J.: Self-attentive sequential recommendation. In: 2018 IEEE international conference on data mining (ICDM). pp. 197–206. IEEE (2018)
19. Kim, D., Park, C., Oh, J., Lee, S., Yu, H.: Convolutional matrix factorization for document context-aware recommendation. In: Proceedings of the 10th ACM conference on recommender systems. pp. 233–240 (2016)
20. Koren, Y., Bell, R.: *Advances in collaborative filtering, recommender systems handbook*, editör: Ricci, f., rokach, l., shapira, b (2015)
21. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* 42(8), 30–37 (2009)
22. Latifi, S., Mauro, N., Jannach, D.: Session-aware recommendation: A surprising quest for the state-of-the-art. *Information Sciences* 573, 291–315 (2021)
23. Lin, J., Pan, W., Ming, Z.: Fissa: fusing item similarity models with self-attention networks for sequential recommendation. In: Proceedings of the 14th ACM Conference on Recommender Systems. pp. 130–139 (2020)
24. Liu, B., Tang, R., Chen, Y., Yu, J., Guo, H., Zhang, Y.: Feature generation by convolutional neural network for click-through rate prediction. In: *The World Wide Web Conference*. pp. 1119–1129 (2019)
25. Liu, H., Lu, J., Yang, H., Zhao, X., Xu, S., Peng, H., Zhang, Z., Niu, W., Zhu, X., Bao, Y., et al.: Category-specific cnn for visual-aware ctr prediction at jd. com. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2686–2696 (2020)
26. Ma, C., Ma, L., Zhang, Y., Sun, J., Liu, X., Coates, M.: Memory augmented graph neural networks for sequential recommendation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 5045–5052 (2020)
27. Ma, Y., Narayanaswamy, B., Lin, H., Ding, H.: Temporal-contextual recommendation in real-time. In: Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2291–2299 (2020)
28. Medsker, L., Jain, L.C.: *Recurrent neural networks: design and applications*. CRC press (1999)
29. Merabet, F.Z., Benmerzoug, D.: Qos prediction for service selection and recommendation with a deep latent features autoencoder. *Computer Science and Information Systems* 19(2), 709–733 (2022)

30. Ouyang, W., Zhang, X., Li, L., Zou, H., Xing, X., Liu, Z., Du, Y.: Deep spatio-temporal neural networks for click-through rate prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2078–2086 (2019)
31. Paterek, A.: Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD cup and workshop. vol. 2007, pp. 5–8 (2007)
32. Pathak, A., Gupta, K., McAuley, J.: Generating and personalizing bundle recommendations on steam. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1073–1076 (2017)
33. Phuong, T.M., Thanh, T.C., Bach, N.X.: Combining user-based and session-based recommendations with recurrent neural networks. In: Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part I 25. pp. 487–498. Springer (2018)
34. Qiu, R., Li, J., Huang, Z., Yin, H.: Rethinking the item order in session-based recommendation with graph neural networks. In: Proceedings of the 28th ACM international conference on information and knowledge management. pp. 579–588 (2019)
35. Quadrana, M., Cremonesi, P., Jannach, D.: Sequence-aware recommender systems. *ACM Computing Surveys (CSUR)* 51(4), 1–36 (2018)
36. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: proceedings of the Eleventh ACM Conference on Recommender Systems. pp. 130–137 (2017)
37. Rendle, S.: Factorization machines. In: 2010 IEEE International conference on data mining. pp. 995–1000. IEEE (2010)
38. Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th international conference on World wide web. pp. 811–820 (2010)
39. Richardson, M., Dominowska, E., Ragno, R.: Predicting clicks: estimating the click-through rate for new ads. In: Proceedings of the 16th international conference on World Wide Web. pp. 521–530 (2007)
40. Ruck, D.W., Rogers, S.K., Kabrisky, M.: Feature selection using a multilayer perceptron. *Journal of Neural Network Computing* 2(2), 40–48 (1990)
41. Ruocco, M., Skrede, O.S.L., Langseth, H.: Inter-session modeling for session-based recommendation. In: Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems. pp. 24–31 (2017)
42. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th international conference on World Wide Web. pp. 285–295 (2001)
43. Seol, J.J., Ko, Y., Lee, S.g.: Exploiting session information in bert-based session-aware sequential recommendation. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2639–2644 (2022)
44. Shen, X., Yi, B., Zhang, Z., Shu, J., Liu, H.: Automatic recommendation technology for learning resources with convolutional neural network. In: 2016 international symposium on educational technology (ISET). pp. 30–34. IEEE (2016)
45. Song, W., Shi, C., Xiao, Z., Duan, Z., Xu, Y., Zhang, M., Tang, J.: Autoint: Automatic feature interaction learning via self-attentive neural networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 1161–1170 (2019)
46. Sun, F., Liu, J., Wu, J., Pei, C., Lin, X., Ou, W., Jiang, P.: Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In: Proceedings of the 28th ACM international conference on information and knowledge management. pp. 1441–1450 (2019)
47. Tang, J., Wang, K.: Personalized top-n sequential recommendation via convolutional sequence embedding. In: Proceedings of the eleventh ACM international conference on web search and data mining. pp. 565–573 (2018)

48. Wan, M., McAuley, J.: Item recommendation on monotonic behavior chains. In: Proceedings of the 12th ACM conference on recommender systems. pp. 86–94 (2018)
49. Wang, S., Cao, L., Wang, Y., Sheng, Q.Z., Orgun, M.A., Lian, D.: A survey on session-based recommender systems. *ACM Computing Surveys (CSUR)* 54(7), 1–38 (2021)
50. Wang, S., Hu, L., Wang, Y., Cao, L., Sheng, Q.Z., Orgun, M.: Sequential recommender systems: challenges, progress and prospects. *arXiv preprint arXiv:2001.04830* (2019)
51. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. pp. 165–174 (2019)
52. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 346–353 (2019)
53. Xu, C., Zhao, P., Liu, Y., Sheng, V.S., Xu, J., Zhuang, F., Fang, J., Zhou, X.: Graph contextualized self-attention network for session-based recommendation. In: *IJCAI*. vol. 19, pp. 3940–3946 (2019)
54. Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., Achan, K.: Self-attention with functional time representation learning. *Advances in neural information processing systems* 32 (2019)
55. Ye, H., Li, X., Yao, Y., Tong, H.: Towards robust neural graph collaborative filtering via structure denoising and embedding perturbation. *ACM Transactions on Information Systems* 41(3), 1–28 (2023)
56. Ying, H., Zhuang, F., Zhang, F., Liu, Y., Xu, G., Xie, X., Xiong, H., Wu, J.: Sequential recommender system based on hierarchical attention network. In: *IJCAI International Joint Conference on Artificial Intelligence* (2018)
57. Yu, F., Liu, Q., Wu, S., Wang, L., Tan, T.: A dynamic recurrent model for next basket recommendation. In: Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval. pp. 729–732 (2016)
58. Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J.M., He, X.: A simple convolutional generative network for next item recommendation. In: Proceedings of the twelfth ACM international conference on web search and data mining. pp. 582–590 (2019)
59. Zhang, M., Wu, S., Gao, M., Jiang, X., Xu, K., Wang, L.: Personalized graph neural networks with attention mechanism for session-aware recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34(8), 3946–3957 (2020)
60. Zhang, Y.: The application of e-commerce recommendation system in smart cities based on big data and cloud computing. *Computer Science and Information Systems* 18(4), 1359–1378 (2021)
61. Zheng, L., Noroozi, V., Yu, P.S.: Joint deep modeling of users and items using reviews for recommendation. In: Proceedings of the tenth ACM international conference on web search and data mining. pp. 425–434 (2017)
62. Zhenzhen, X., Jiang, H., Kong, X., Kang, J., Wang, W., Xia, F.: Cross-domain item recommendation based on user similarity. *Computer Science and Information Systems* 13(2), 359–373 (2016)

Shangzhi Guo received the B.Sc. degree in software engineering from Hunan Normal University in 2005, the M.Sc. degree in software engineering from Hunan University in 2018. He is currently studying for a Ph.D. in mechanical engineering from Chongqing University as a senior engineer. His research interests are artificial intelligence and intelligent manufacturing, intelligent recommendation, system architecture, and big data application.

Xiaofeng Liao received the B.Sc. degree in computational mathematics from Sichuan University in 1986, the M.Sc. degree in computational mathematics from Sichuan University in 1992, Ph.D. in circuits and systems from University of Electronic Science and Technology of China in 1997, IEEE Fellow, AIAA Fellow. His research interests include computational intelligence and information security, artificial neural networks, dynamic theory, cryptography.

Fei Meng received the B.Sc. degree in physical technology from Huaqiao University in 2012, the M.Sc. degree in computer technology engineering from Chongqing University in 2018. She is currently studying for a Ph.D. in transportation from Chongqing University. She research interests include spatiotemporal trajectory data processing and big data application.

Qing Zhao received the B.Sc. degree in electrical engineering from Chongqing University in 2001, the M.Sc. degree in software engineering from Fudan University in 2009. He is currently studying for a Ph.D. in mechanical engineering from Chongqing University. His research interests include artificial intelligence and intelligent manufacturing, data security, network security, and big data applications.

Yuling Tang received the B.Sc. degree in computer science and technology from Tianjin Vocational and Technical Normal University in 2010. She research interests include medical big data, auxiliary medicine, and smart medicine.

Hui Li received the B.Sc. degree in management from Jiangxi Normal University in Nanchang, China in 2019, the M.Sc. degree in computer technology from Jiangxi Normal University in 2022. He is currently working on data integration and mining at the Jiangxi Provincial Institute of Land and Space Survey and Planning, Nanchang, China. His research interests includes natural language processing and data mining.

Qinqin Zong received the B.Sc. degree in software engineering from Jiangxi Normal University in 2018, the M.Sc. degree in computer science and technology from Jiangxi Normal University in 2021. She is currently working at Jiangxi Biotechnology Vocational College. She research interests include natural language processing and recommendation systems.

Received: May 22, 2023; Accepted: August 20, 2023.

Applying SPIN Checker on 5G EAP-TLS Authentication Protocol Analysis

Qianli Wang

Software School,
East China JiaoTong University,
Nanchang, China
qianjustice2@163.com

Abstract. Currently, there is relatively little formal analysis and verification work on the 5G EAP-TLS authentication protocol. In this paper, we use the model checker SPIN to perform a formal analysis of the 5G EAP-TLS authentication protocol. Firstly, we analyze the process of the 5G EAP-TLS authentication protocol and abstract it to obtain a formal model of the protocol. Then, we describe the construction of the protocol model based on the Promela language. The unique feature of this paper is the replacement of the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted by an unknown subject public key. This is because the Promela language in SPIN has an eval function that can check the value of each field. This can replace the function of the hash function and make the Promela model construction more portable. The paper analyzes the attack paths of the protocol and reveals design flaws that undermine the expected identity authentication attributes and secret consistency of the protocol. The results not only provide a comprehensive understanding of the security properties of the 5G EAP-TLS authentication protocol but also offer valuable insights and guidance for the verification of the protocol's security properties, security design, and optimization of protocol implementation and interoperability.

Keywords: 5G EAP-TLS; SPIN; Formal Analysis; Model Checking.

1. Introduction

The introduction of 5G technology will undoubtedly change the way we use the Internet and communicate, but it also brings risks and challenges in terms of network security. Therefore, 5G users are not only concerned about the improvement of network speed, but also particularly concerned about ensuring network security [1]. 5G networks are constantly facing security issues such as eavesdropping and surveillance, device intrusion, privacy data leakage, honeypot attacks, and network fraud. In order to alleviate these risks and challenges, 5G networks need to adopt a series of security measures, such as encryption technology, authentication and access control of devices, encrypted storage and transmission of data, authentication and access control, vulnerability management, and security training.

In the 5G network, three identity authentication protocols are defined in the relevant 3GPP documents, including the 5G AKA (Authentication and Key Agreement) protocol

[2], the EAP-AKA protocol [2], and the 5G EAP-TLS protocol [2,3]. The first two protocols use symmetric key passwords and are protocols used for authentication and key negotiation. The 5G EAP-TLS protocol uses public key passwords and is a certificate-based authentication protocol. The 5G EAP-TLS protocol is a certificate-based authentication protocol that uses digital certificates to verify the user's identity. In this protocol, the authentication between the client and server is based on the certificates they hold, which are signed by a trusted third-party certificate authority (CA). The identity authentication and key negotiation methods provided by these three protocols are different, so their specific application scenarios are also different. So far, these protocols are still in the standardization process and are mainly developed and released in the form of RFCs. The main sources of serious security vulnerabilities in the implementation of informal protocols are protocol design issues and implementation problems, for example, as reported in literature [4,5]. Formal analysis is an effective method to solve the ambiguity and verify the correctness of the protocol design. First, the protocol model is constructed using a formal language such as Promela, and then the formal protocol model is analyzed to see if it meets the required security properties. Symbolic model checking [6] is an important method for performing formal analysis of protocols. Since the first work [7] using this technique to analyze the design flaws of the Needham-Schroeder protocol, the symbolic model checking of security protocols has been a hot research field, it has been considered a powerful technique for formal analysis of security protocol [8-14] design, and has been applied in some real-world Protocols, such as TLS [15].

2. Related Work

Zhang Jingjing's work [1] built a formal model for the 5G EAP-TLS authentication protocol based on pi calculus and used the ProVerif model checker to formally verify the model, revealing several weaknesses and design flaws in the current protocol, thereby undermining the expected identity authentication attributes. Her work provides strong support for the formal verification of the 5G EAP-TLS authentication protocol. Chen Liping [16] studied the formal model of the EAP-TLS protocol and security attributes based on ProVerif, and verified the mutual authentication between user devices and the network, as well as the confidentiality of the security anchor key KSEAF and the user identity identifier SUPI in the protocol. Wang Yuedong [17] constructed a protocol interaction model based on the Diffie-Hellman pattern, then extended the Dolev-Yao attacker model, proposed two controlled participant scenarios and a mixed channel model, and finally used the verification tool SmartVerif to verify the confidentiality, authentication, and privacy of the protocol. This research provides valuable references for designing secure communication protocols. However, the formal analysis and verification of the 5G EAP-TLS authentication protocol, as a novel and important protocol, is relatively scarce, and it is urgently needed for scholars to analyze and verify it from different perspectives, different methods, and even different tools to better maintain the security of the network in the 5G environment.

SPIN [18], developed by Holzmann, is a general model checking tool used to verify logical consistency in concurrent systems. It is loved by scholars for its concise

modeling language, support for Linear Temporal Logic (LTL), and good algorithmic structure. The literature [19] uses SPIN to verify known vulnerabilities in the classic NSPK cryptographic protocol and proposes a static analysis message construction method to mitigate the state explosion problem, providing guidance for future scholars using SPIN for security protocol verification. Since then, a large number of researchers have used SPIN to formally verify security protocols and have achieved good results [20-23]. Therefore, this paper uses the model checker SPIN [24-25] to perform formal analysis and verification of the 5G EAP-TLS authentication protocol [26].

The main contributions of this paper are twofold: First, it replaces the hash value of the 5G EAP-TLS authentication protocol with a message content field encrypted using an unknown subject public key. This is because the Promela language in SPIN has an "eval" function that can detect the value of each field, which can replace the functionality of the hash function. The advantage is that it makes the Promela model construction more portable. Second, based on the formal model in literature [1], this paper uses the model checker SPIN to perform formal analysis of the 5G EAP-TLS authentication protocol and verifies the confidentiality, authenticity, and secret consistency of the user identity SUPI, the pre-master key Rprekey, and the master key Kseaf.

The rest of this paper is organized as follows. Section 2 provides a detailed introduction to the abstract model of the 5G EAP-TLS protocol, and describes the process of constructing the protocol model based on the Promela language, and implements the security properties of the 5G EAP-TLS authentication protocol. In Section 3, the results of SPIN verification are presented, and the attack paths of the protocol are discussed. Section 4 discusses the results obtained. Finally, Section 5 summarizes the work of this study and outlines future directions.

3. Research Methods

3.1. Abstraction Of 5G EAP-TLS Authentication Protocol

Based on the protocol flow described above, this section will abstract the 5G EAP-TLS authentication protocol. The formalized model of the abstracted protocol is shown in Fig.1. Since the message transmission between the service network and the home network is usually wired and forwarded, it can be assumed that the message transmission between them is secure. Therefore, this article abstracts the 5G EAP-TLS authentication protocol as an interaction between two roles: user terminal A and network B, where network B represents a combined module of the service network and the home network.

First, the user terminal encrypts the identity SUPI and a newly generated random number N_a using the public key of network B, and sends the message to network B. Upon receiving the message, network B sends an EAP-TLS protocol start signal Start to the terminal A, and the terminal generates a new random number N_{a1} and sends it back to the network. After receiving the response from the terminal, network B generates its own random number N_b and its certificate and sends them to the terminal. The

certificate is represented as $\{B\}SK_b$ in this article. The terminal verifies the certificate first, and then generates a pre-master key K_{ab} and a master key K_{seaf} , where K_{seaf} is calculated from N_{a1} , N_b , and K_{ab} . Since N_{a1} and N_b are transmitted in plaintext, the secrecy of K_{seaf} will be guaranteed by K_{ab} . The terminal calculates the hash value of the previous two message contents and sends its own certificate, the signed hash value, and the hash value encrypted with the master key K_{seaf} to network B. Note that since the hash function is a one-way encryption function and has anti-inverse property, a public key of an unknown subject can be used to simulate hash encryption in the formalization process. This is because anyone can encrypt a message using this public key, but the private key corresponding to this public key is secret, so no one can reverse the hashing process. To simplify the model, this article uses the Hash function to calculate the hash value used for signature, and uses the message encrypted with the unknown subject public key PK_z to replace the hash value encrypted with the master key.

After receiving the message, the network side verifies the signature of the terminal A, calculates the master key K_{seaf} based on the second received random number N_{a1} , its own random number N_b , and the received pre-master key K_{ab} . The network side then decrypts the message using K_{seaf} , compares it with its own first two messages after performing a hash operation, and if the hash value matches the received hash value, the network side successfully verifies the identity of terminal A. The network B then encrypts its third and fourth messages using an unknown principal's public key PK_z , and then encrypts them using K_{seaf} before sending them to terminal A. After receiving the message, terminal A decrypts it using K_{seaf} and compares the decrypted message to verify the identity of the network side.

Unlike the formal model in the literature [1], this article replaces one hash value in the fifth message and the hash value in the sixth message with the encrypted message content field using an unknown principal's public key PK_z . This is because the SPIN Promela language has an eval function that can check the value of each field, which can replace the function of the hash function and make the construction of the Promela model more convenient.

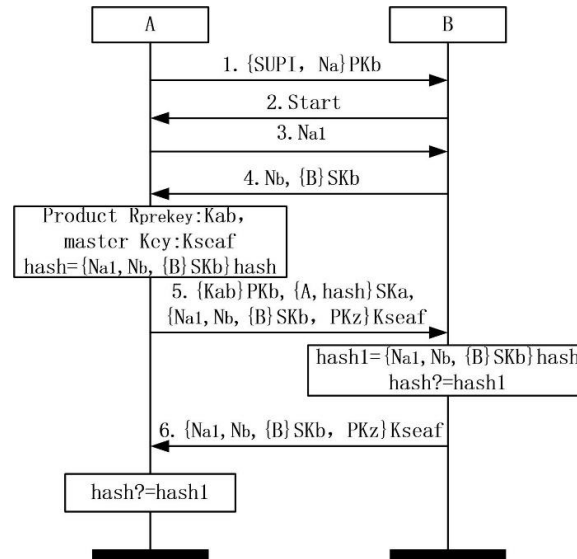


Fig. 1. Abstracted 5G EAP-TLS authentication protocol

3.2. Model Assumptions

The formal model of a security protocol is typically described as a collection of honest entities that interact with each other according to the protocol rules, and an intruder who is powerful enough to perform all possible attack behaviors. The communication network between the entities is abstracted as the protocol message channels.

To better reveal the security vulnerabilities in the protocol, rather than the weaknesses of the cryptographic system used by the protocol, we usually make perfectness assumptions about the underlying cryptographic algorithms of the protocol, as follows:

- Only the entity that knows the corresponding decryption key can decrypt a message.

- It is impossible to derive the encryption key from the encrypted message itself.

- There is sufficient redundancy in the protocol messages to enable the decryption algorithm to detect whether the message is encrypted with the expected key.

- The encryption components cannot be tampered with, and it is not possible to forge a new large encryption component using multiple encryption components.

- The intruder model in this paper is based on the well-known Dolev-Yao model, which provides an important principle: never underestimate the capabilities of the intruder. Therefore, we always assume that the intruder is very powerful, able to monitor the entire communication network for eavesdropping, interception, analysis, forgery, and other behaviors, and can also participate in the protocol interaction as a normal entity. In addition, the intruder can continually learn knowledge during the protocol execution and use it to attack the normal protocol process. To better represent the capabilities of the

intruder, this paper assumes that honest entities cannot communicate directly with each other, and that all messages will be intercepted by the intruder.

The construction of a security protocol model based on SPIN can be roughly divided into two steps: modeling the honest entities and modeling the intruder. The following sections will provide specific introductions.

3.3. Honest Entity Modeling

The first step in modeling the 5G EAP-TLS protocol is to define a finite set of names for the protocol model. The Promela definition is as follows:

```
mtype = {NULL,SUPI,Start,Na,Na1,Nb,A,B,I,
R, gD,PKa,PKb,PKi,PKz,SKa,SKb,SKi,Kab,
Kai,Kseaf,Kseaf1};
```

NULL represents a placeholder, SUPI is the unique identifier of the initiator, Na, Na1, and Nb are random numbers generated by honest parties, A and B are the identity tags of honest parties, I is the identity tag of the intruder, R is the service constant of the intruder, gD is the generic data of the intruder, PKa, PKb, and PKi represent the public keys of the parties, PKz represents the public key of an unknown party, SKa, SKb, and SKi represent the private keys of the parties, Kab represents the session key between honest parties A and B, and Kai represents the session key known by the intruder.

Next, I define the communication channels for protocol interaction. Considering the length of the message field in the protocol, I define three synchronous channels with lengths of 4, 12, and 7, respectively. The specific Promela implementation is as follows:

```
chan ca = [0] of {mtype, mtype, mtype, mtype};
chan cb = [0] of {mtype, mtype, mtype,mtype,
mtype, mtype, mtype,mtype, mtype, mtype,
mtype, mtype};
chan cc = [0] of {mtype, mtype, mtype,mtype,
mtype, mtype, mtype}
```

The channel ca is used to transmit messages 1, 2, 3, and 4, channel cb is used to transmit message 5, and channel cc is used to transmit message 6. For honest parties, the first field of the message is filled with the sender's identity. Any remaining fields are filled with NULL.

After abstraction, the 5G EAP-TLS protocol involves two honest parties. Therefore, this article defines the initiator process PIni and the responder process PRes to respectively depict the behaviors of the two roles in the protocol, and instantiates the parameters. The initiator process has four process parameters, which are self, party, nonce1, and nonce2. Among them, the parameter "self" indicates the subject of the initiator process itself, "party" indicates the expected communication subject of the initiator process, "nonce1" indicates the first random number generated by the initiator, and "nonce2" indicates the second random number generated by the initiator random number. According to the abstracted protocol specification, there are 6 communication

statements in the initiator process. In order to reduce the number of statement interactions between processes and better reduce the state, the "atomic" keyword is used to combine the message receiving statement with the next message Send statements are combined into one atomic step. The process performs message type matching and simulates the function of the hash function by using the "eval(x)" function to determine whether the received value is equal to x. As mentioned above, in order to simplify the message field length, we use the inline function "Hash" in the process to hash the initiator's previous message content, and update the value of the global variable "hash", thereby replacing the complete message 5. The first hash function in the field.

In addition, some macros and inline functions used to describe the nature of the protocol are included in the initiator process, including: "IniRunning" and "IniCommit" are used to represent the initiation and commit phases of the protocol between the initiator and the responder; the inline function "Secrecy" is an assertion used to determine whether the confidentiality of the session key "Kab" has been violated; the inline function "IniSec" is used to update the value of the global variable "AiniSec", which is used to store the session secret obtained by the initiator key. The specific implementation will be described in detail in the section on security attributes.

The responder process is similar to the initiator process. The process parameters include "self" representing the responder process itself and "nonce1" representing the responder's random number. According to the protocol specification, the responder needs to compare the received hash value with the hash of the previous messages it participated in. If the hash values are equal, the responder process can continue to execute. The specific implementation of the responder process will not be described in detail.

Finally, an initialization process needs to be defined to instantiate the initiator and responder processes. The instantiation of the main processes is represented using process instance statements in this process.

The initiator process can choose to communicate with the subject B or with the intruder I, while the responder process only needs to respond to protocol session requests from other subjects and does not need to select communication partners. The PI process is the intruder process and will be introduced later.

3.4. Intruder Modeling

The intrusion message construction strategy in this article adopts the static analysis method in reference [19], which can reduce the number of protocol states and obtain a simpler model by manually screening out invalid messages forged by intruders.

First, I define a set of local variables x_1 to x_{10} to receive the message fields intercepted by the intruder. Secondly, the intruder's knowledge representation is restricted by the static analysis method to define the intruder's initial knowledge set. This article defines the intruder's initial knowledge as subject identity, relevant public keys, the intruder's own key and generic data, including A, B, I, PKa, PKb, PKi, SKi, Kai, gD. When the intruder intercepts a message, it can decompose and learn from the message to increase its knowledge. If the message cannot be decrypted, the entire encryption component is learned. Tab.1 shows all the messages that the intruder may intercept and the knowledge items that can be learned.

The first column in Tab.1 shows all the messages that the intruder may intercept, and the second column shows the knowledge that the intruder can learn through intercepting the messages. Tab.2 lists the messages that the intruder can forge and the knowledge required to send the messages.

Table 1. Knowledge that an intruder can ultimately obtain

Received message	Learning knowledge
{SUPI,Na}PKb;	{SUPI,Na}PKb;
{SUPI,Na}PKi;	SUPI; Na; Start; Na1;
Start; Na1; gD; Nb;	Nb; {B}SKb;
Nb, {B}SKb;	{Kab}PKb;
{Kab}PKb, {A,hash}SKa,	{A,Na1,Nb, {B}SKb}S
{{Na1,Nb, {B}SKb},pkz}	Ka;
Kseaf;	{{Na1,Nb, {B}SKb}P
{Kab}PKb, {A,hash}SKa,	Kz} Kseaf;
{{Na1,Nb, {B}SKb}PKz}	{A,Na1,gD, {B}SKb}SKa;
Kseaf;	{{Na1,gD, {B}SKb},PKz}
{Kab}PKb, {A,hash}SKa,	Kseaf;
{{Na1,gD, {B}SKb}PKz}	{{Na1,Na, {B}SKb}PKz}
Kseaf;	Kseaf;
{Kai}PKi, {A,hash}SKa,	{A,Na1,Na, {B}SKb}SKa;
{{Na1,Nb, {B}SKb}PKz}	{{Na,Nb, {B}SKb}PK
Kseaf1;	z} Kseaf;
{Kai}PKi, {A,hash}SKa,	{{gD,Nb, {B}SKb}PKz}
{{Na1,Na, {B}SKb}PKz}	Kseaf;
Kseaf1;	{Na1,Nb, {B}SKb}PK
{Kai}PKi, {A,hash}SKa,	z
{{Na1,gD, {B}SKb}PKz}	
Kseaf1;	
{{Na1,Nb, {B}SKb}PKz}	
Kseaf;	
{{Na,Nb, {B}SKb}PKz}	
Kseaf;	
{{gD,Nb, {B}SKb}PKz}	
Kseaf;	

The intersection of the knowledge that the intruder may obtain and the knowledge that the intruder needs is the truly useful knowledge for the intruder.

The intersection is shown below:

- Atomic message: Na; Start; Na1; Nb;SUPI;

- Composite message:

{SUPI,Na}PKb; {B}SKb; {Kab}PKb;

{A,Na1,Nb, {B}SKb}SKa;

{{Na1,Nb, {B}SKb}PKz}Kseaf;

{{gD,Nb, {B}SKb}PKz}Kseaf;

{{Na,Nb, {B}SKb}PKz}Kseaf;

{{Na1,gD, {B}SKb}PKz}Kseaf;

{Na1,Nb, {B}SKb}PKz;

Table 2. Knowledge that an intruder may need

Send Message	Required knowledge
{SUPI,Na}PKb;{I,Na}Pkb;	SUPI;
{SUPI,Nb}PKb;	Na;
{SUPI,gD}PKb;{I,gD}PKb;	NaI;
Start;NaI:gD;Na;	Nb;
Nb,{B}SKb;gD,{B}SKb;	{SUPI,Na}PKb;
Nb,{I}SKi;gD,{I}SKi;	{SUPI,Nb}PKb;
{Kab}PKb,{A.hash}SKa,	Start;
{{NaI,Nb,{B}SKb}PKz}Kseaf;	{B}SKb;
{Kai}PKb,{A.hash}SKa,{{NaI,Nb,	{Kab}PKb;
{B}SKb}PKz}KseafI;	Kab;
{Kab}PKb,{A.hash}SKa,{{Na,Nb,	Kseaf;
{B}SKb}PKz}Kseaf;	{A,NaI,Nb,{B}SKb}S
{Kai}PKb,{A.hash}SKa,{{Na,Nb,	Ka;
{B}SKb}PKz}KseafI;	{{NaI,Nb,{B}SKb}
{Kab}PKb,{A.hash}SKa,{{gD,Nb,	PKz}Kseaf;
{B}SKb}PKz}Kseaf;	{A,Na,Nb,{B}SKb}S
{Kai}PKb,{A.hash}SKa,{{gD,Nb,	Ka;
{B}SKb}PKz}KseafI;	{{Na,Nb,{B}SKb}
{Kab}PKb,{I,hash}SKi,{{NaI,Nb,{B}S	PKz}Kseaf;
Kb}PKz}Kseaf;	{A,gD,Nb,{B}SKb}S
{Kai}PKb,{I,hash}SKi,{{NaI,Nb,	Ka;
{B}SKb}PKz}KseafI;	{{gD,Nb,{B}SKb}
{Kab}PKb,{I,hash}SKi,{{Na,Nb,	PKz}Kseaf;
{B}SKb}PKz}Kseaf;	{{NaI,gD,{B}SKb}
{Kai}PKb,{I,hash}SKi,{{Na,Nb,	PKz}Kseaf;
{B}SKb}PKz}KseafI;	{{NaI,gD,{I}SKi}
{Kab}PKb,{I,hash}SKi,{{gD,Nb,	PKz}Kseaf;
{B}SKb}PKz}Kseaf;	{NaI,Nb,{B}SKb}
{Kai}PKb,{I,hash}SKi,{{gD,Nb,	PKz
{B}SKb}PKz}KseafI;	
{NaI,Nb,{B}SKb}PKz}Kseaf;	
{{NaI,gD,{B}SKb}PKz}Kseaf;	
{{NaI,Nb,{B}SKb}PKz}KseafI;	
{{NaI,gD,{B}SKb}PKz}KseafI;	
{I,Nb}PKb;	
{{NaI,Nb,{I}SKi}PKz}Kseaf;	
{{NaI,gD,{I}SKi}PKz}Kseaf;	
{{NaI,Nb,{I}SKi}PKz}KseafI;	
{{NaI,gD,{I}SKi}PKz}KseafI;	

First, it is necessary to initialize the knowledge items of the intruder using a bit-type variable based on the intersection obtained. When the intruder learns a certain knowledge item, update the value of the corresponding variable to 1. The intruder process behaves as an endless process, constantly receiving or sending messages from to the channel, which can be achieved by a do loop. Then construct the intruder's sending message statements based on Tab.2 and the content of the intersection, and construct the intruder's receiving statements using the intersection. The update of the intruder's knowledge is implemented through macros k1, k2, k3, and k5. The receive statements and the corresponding knowledge learning are both placed in d_step statements, and at the end of the statements, the variables x1, etc. need to be reset to zero, which helps to reduce the number of system states.

3.5. Security Attributes

This paper verifies the authentication, confidentiality and secret consistency of the 5G EAP_TLS protocol. Authentication requires mutual authentication of identities between the terminal and the network after the execution of the protocol is completed. This paper is based on the method of reference [19] to characterize weak consistency through LTL linear temporal logic. The specific implementation is as follows:

```
ltl e1 {([] (([]!r) || (!r U s))&&(([]!P) || (!p U q)))}
```

The specific meaning is that when role A initiates a protocol session with role B, role B must have participated in a session with role A before this. The reverse implication is also true. Macros such as IniRunningAB are used to update the values of the corresponding global variables.

Confidentiality of the 5G EAP_TLS protocol requires that the user terminal's unique identity SUPI, newly generated pre-master key Kab, and master key Kseaf will not be learned by intruders before the protocol execution between the user terminal and the network ends. This article uses the Secrecy macro to implement the protocol confidentiality assertion.

On the premise that the intruder does not participate in protocol interaction as an honest entity, the intruder cannot obtain Kab, Kseaf and SUPI by attacking the protocol.

The secret consistency of the 5G EAP_TLS protocol requires that at the end of the protocol execution between the user terminal UE and the network AUSF, the two parties should agree on the pre-master key Rprekey. In the abstract model of this article, this means that the main keys of roles A and B reach an agreement, which is implemented through LTL. The Promela implementation is as follows:

```
ltl e2 {([] ((AiniSec != NULL && BresSec !=  
NULL) -> (AiniSec == BresSec)))}
```

4. Experiment and Result

The experimental environment of this article is as follows: Intel i5 CPU, 64-bit Linux, 2G RAM, and Spin V6.5.1. In depth-first search mode, Spin is used to verify the authentication, confidentiality, and secret consistency of the 5G EAP-TLS protocol, with a maximum search depth of the default value 10000.

First, the authentication of the 5G EAP-TLS protocol is verified, and a loophole was found.

The attack path that violates the authentication is shown in Fig.2.

The specific attack steps are shown below:

Step 1: The initiator A sends SUPI and random number Na to the responder B, and encrypts them using the responder's public key PKb.

```
ca ! A, SUPI, Na,PKb;
```

Step 2: The intruder I intercepts the message and masquerades as responder B to send an EAP-TLS start message "Start" to the initiator A.

```
ca ! A, Start, NULL,NULL;
```

Step 3: The initiator A receives the message and sends a new random number Na1 to responder B.

ca ! A, Na1, NULL, NULL;

Step 4: The intruder intercepts the message and masquerades as responder B to send its own generic data gD and the certificate {B}SKb obtained from the previous session to initiator A.

ca ! A, gD, B, SKb;

Step 5: The initiator A receives the message, verifies the certificate, and then sends the pre-master secret Kab encrypted with B's public key, its own certificate, the hash value of the first two message contents signed, and the first two message contents encrypted with the master key using PKz.

cb ! A, Kab, PKb, A, hash, Ska, Na1, gD, B,
SKb, PKz, Kseaf;

Step 6: The intruder intercepts the message and captures the content of the message encrypted with the master key, then masquerades as responder B and sends it to initiator A. After receiving the message, A verifies the message fields and completes the authentication of B's identity.

cc ! Na1, gD, B, SKb, PKz, Kseaf;

As can be seen from the above attack path, the initiator A believes that it has completed mutual authentication with the responder B, but in reality, the responder B did not participate in the protocol execution with A, which violates the authenticity.

Next, use SPIN to verify the confidentiality of SUPI, pre-master key Kab, and master key Kseaf in the 5G EAP-TLS protocol. The verification results show that the confidentiality of SUPI, Kab, and Kseaf in the 5G EAP-TLS protocol can be satisfied, and no loopholes have been found.

Finally, SPIN is used to verify the secrecy consistency of the 5G EAP-TLS protocol, and a vulnerability was found.

The attack sequence that violates the secrecy consistency is shown in Fig.3, and the specific attack steps are shown below:

Step 1: The initiator A sends SUPI and a random number Na to the responder B, and encrypts them using the responder's public key PKb.

ca ! A, SUPI, Na, PKb;

Step 2: The intruder I intercepts the message and forwards it to the responder B.

ca ! B, SUPI, Na, PKb;

Step 3: The responder B receives the message and verifies SUPI, then sends a protocol start message "Start" to the initiator A.

ca ! B, Start, NULL, NULL;

Step 4: The intruder intercepts the message and forwards it to the initiator A.

ca ! A, Start, NULL, NULL;

Step 5: The initiator A receives the message and sends a new random number Na1 to the responder B.

ca ! A, Na1, NULL, NULL;

Step 6: The intruder intercepts the message and forwards it to the responder B.

ca ! B, Na1, NULL, NULL;

Step 7: The responder B receives the message and sends its own random number Nb and certificate to the initiator A.

ca ! B, Nb, B, SKb;

Step8: The intruder intercepts the message and forwards it to the initiator A.

ca ! A, Nb, B, SKb;

Step 9: The initiator A receives the message, verifies the received certificate, and if verification succeeds, sends the pre-shared key Kab encrypted with B's public key, its own certificate, the hash value of the first two message contents signed, and the first two message contents encrypted with PKz that has been used with the master key.

cb ! A, Kab, PKb, A, hash, Ska, Na1, gD, B,
SKb,PKz, Kseaf;

Step 10: The intruder intercepts the message, modifies the pre-shared key Kab to Kai, forges a new master key Kseaf1 using the modified pre-shared key Kab, Na1, and Nb, and sends it to the responder B in the message format.

cb ! B, Kai, PKb, A, hash, Ska, Na1, Nb, B,
SKb, PKz, Kseaf1;

Step 11: The responder B receives the message, verifies the hash value, treats Kai as the pre-shared key and Kseaf1 as the master key, encrypts the first two message contents that have been encrypted with PKz using Kseaf1, and sends them to the initiator A.

cc ! B, Na1, Nb, B, SKb,PKz, Kseaf1;

Step 12: The intruder intercepts the message, uses fragments of previously intercepted messages to forge a message, and sends it to the initiator A. The initiator A receives the message, verifies it, and upon successful verification, considers itself to have achieved agreement with the responder B on the pre-shared key and the master key.

cc ! A, Na1, Nb, B, SKb, PKz, Kseaf;

As can be seen from the above attack steps, the initiator A generates a pre-master key Kab and a master key Kseaf, while the responder receives a pre-master key Kai and a master key Kseaf1. However, both parties believe that they have reached a consensus on the values of the pre-master and master keys. Therefore, the 5G EAP-TLS protocol violates the secret consistency

5. Discussion

The results of this paper provide valuable insights into the security of the 5G EAP-TLS authentication protocol from both theoretical and practical perspectives. The formal model of the protocol constructed based on the Promela language can be used to verify whether the protocol satisfies properties such as authentication, secrecy, and secret consistency, and to discover possible vulnerabilities in the protocol. The use of the model checker SPIN to perform formal analysis and verification of the protocol provides a reliable and effective method for ensuring the security of the 5G network environment.

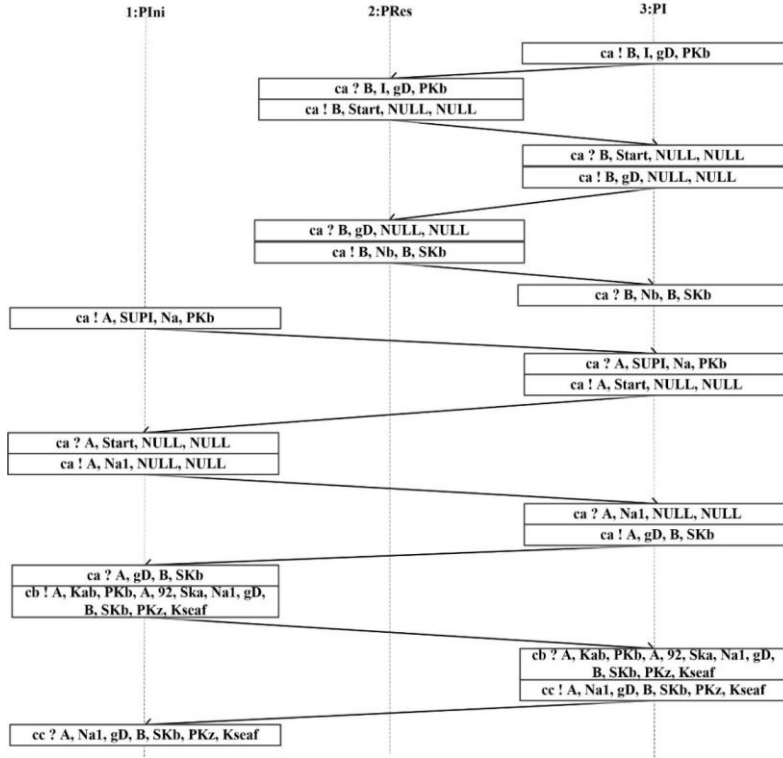


Fig. 2. Attack path for authentication violation of 5G EAP-TLS protocol

The unique feature of this paper is the replacement of the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted by an unknown subject public key, which not only makes the construction of the Promela model more convenient but also provides more comprehensive coverage of possible attack paths. In addition, the paper analyzes the attack paths of the protocol and reveals design flaws that undermine the expected identity authentication attributes and secret consistency of the protocol. This could serve as a reference for the future design and implementation of secure communication protocols in 5G networks.

In summary, The SPIN model can abstract the EAP-TLS protocol into a state machine model, perform formal verification on the protocol process, and discover potential security vulnerabilities. This is beneficial for theoretically optimizing and repairing the EAP-TLS protocol. The SPIN model can automate testing of the EAP-TLS protocol, detect abnormal behavior during actual deployment and operation, and discover unknown security attack points. Once a security vulnerability is detected, the SPIN model can provide replication steps and repair suggestions to help developers fix the vulnerability in a timely manner and improve the security of 5G networks. The SPIN model detection results can serve as a test report for the 5G device certification function, providing reference for device certification and improving product quality. Formal validation of the EAP-TLS protocol can provide stricter specifications, which can help unify and optimize the differences between EAP-TLS implementations from different

vendors, and improve interoperability, the results of this paper not only provide a comprehensive understanding of the security properties of the 5G EAP-TLS authentication protocol but also offer valuable insights and guidance for the design and verification of secure communication protocols in 5G networks.

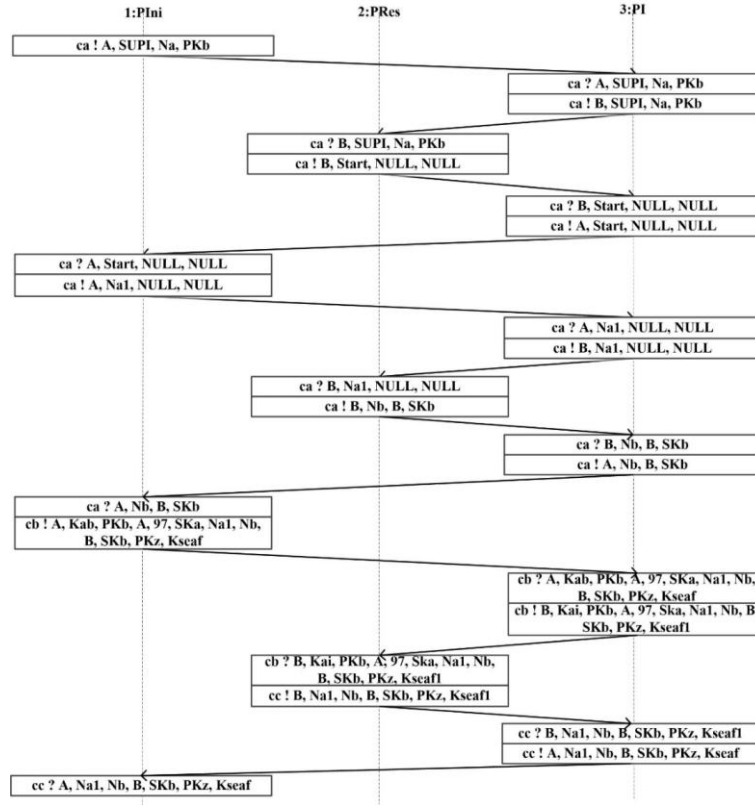


Fig. 3. Attack path for secret consistency violation of 5G EAP-TLS protocol

6. Conclusion

In this paper, a model detector SPIN is used to formally analyze the security properties of the 5G EAP-TLS authentication protocol. The special thing is that this article replaces the hash value of the 5G EAP-TLS authentication protocol with the message content field encrypted with the public key of the unknown subject. This is because there is an eval function in the Promela language of SPIN, which can detect the value of each field. This can replace the function of the hash function, and the advantage is that it can make the construction of the Promela model more convenient. This paper describes the 5G EAP-TLS authentication protocol process, abstracts the authentication protocol, and obtains the formal model of the 5G EAP-TLS authentication protocol. Then, the

protocol model construction based on Promela language is described in detail, the honest subject modeling and intruder modeling are constructed, and the weak consistency and secret consistency of the protocol are described by LTL linear temporal logic. Finally, the authentication and secret consistency of the protocol are verified. The results of verification and analysis reveal several design flaws, indicating that there are indeed loopholes in the identity authentication attributes and secret consistency of the 5G EAP-TLS authentication protocol, and the attack path has been run out.

In view of the defects of 5G EAP-TLS, the protocol will be improved next, and the improved protocol will be verified. At the same time, considering that the new protocol is becoming more and more complex, our next step is to improve the Promela modeling method to alleviate the state explosion problem.

Acknowledgment. The research was supported by Innovation and Entrepreneurship Training Program for College Students in Jiangxi Province.

References

1. Zhang, J., Yang, L., Cao, W., & Wang, Q. (2020). Formal analysis of 5G EAP-TLS authentication protocol using proverif. *IEEE access*, 8, 23674-23688. <https://doi.org/10.1109/ACCESS.2020.2969474>
2. 3GPP. (2020). Security architecture and procedures for 5G system. Technical Specification (TS) 3GPP TS 33.501 V17. 0.0 (2020–2012).
3. Dierks, T., & Rescorla, E. (2008). The transport layer security (TLS) protocol version 1.2 (No. rfc5246). <https://doi.org/10.17487/RFC5246>
4. Basin, D., Cremers, C., Miyazaki, K., Radomirovic, S., & Watanabe, D. (2014). Improving the security of cryptographic protocol standards. *IEEE Security & Privacy*, 13(3), 24-31. <https://doi.org/10.1109/MSP.2013.162>
5. Hirschi, L., Sasse, R., & Dreier, J. (2019). Security issues in the 5G standard and how formal methods come to the rescue. *ERCIM News*. <https://hal.science/hal-02268822>
6. Basin, D., Cremers, C., & Meadows, C. (2018). Model checking security protocols. *Handbook of Model Checking*, 727-762. https://doi.org/10.1007/978-3-319-10575-8_22
7. Lowe, G. (1995). An attack on the Needham–Schroeder public–key authentication protocol. *Information processing letters*, 56(3). [https://doi.org/10.1016/0020-0190\(95\)00144-2](https://doi.org/10.1016/0020-0190(95)00144-2)
8. Mitchell, J. C., Mitchell, M., & Stern, U. (1997, May). Automated analysis of cryptographic protocols using mur/spl phi. In *Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No. 97CB36097)* (pp. 141-151). IEEE. <https://doi.org/10.1109/SECPRI.1997.601329>
9. Song, D. X. (1999, June). Athena: a new efficient automatic checker for security protocol analysis. In *Proceedings of the 12th IEEE Computer Security Foundations Workshop (pp. 192-202)*. IEEE. <https://doi.org/10.1109/CSFW.1999.779773>
10. Clarke, E. M., Jha, S., & Marrero, W. (2000). Verifying security protocols with Brutus. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 9(4), 443-487. <https://doi.org/10.1145/363516.363528>
11. Armando, A., & Compagna, L. (2004). SATMC: a SAT-based model checker for security protocols. In *Logics in Artificial Intelligence: 9th European Conference, JELIA 2004, Lisbon, Portugal, September 27-30, 2004. Proceedings 9* (pp. 730-733). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-30227-8_68
12. Basin, D., Mödersheim, S., & Vigano, L. (2005). OFMC: A symbolic model checker for security protocols. *International Journal of Information Security*, 4, 181-208. <https://doi.org/10.1007/s10207-004-0055-7>

13. Cortier, V., Delaune, S., & Lafourcade, P. (2006). A survey of algebraic properties used in cryptographic protocols. *Journal of Computer Security*, 14(1), 1-43.
<https://doi.org/10.3233/JCS-2006-14101>
14. Blanchet, B. (2016). Modeling and verifying security protocols with the applied pi calculus and ProVerif. *Foundations and Trends® in Privacy and Security*, 1(1-2), 1-135.
<http://dx.doi.org/10.1561/33000000004>
15. Cremers, C., Horvat, M., Hoyland, J., Scott, S., & van der Merwe, T. (2017, October). A comprehensive symbolic analysis of TLS 1.3. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1773-1788).
<https://doi.org/10.1145/3133956.3134063>
16. Chen Liping, Xu Peng, Wang Danchen & Xu Yang. (2022). Formal Verification Research of EAP-TLS Protocol. *Computer Science* (S2), pp. 685-689.
<https://doi.org/10.11896/jsjx.211100111>
17. Wang Yuedong, Xiong Yan, Huang Wenchao & Wu Jianshuang. (2021). A Formal Analysis Scheme for 5G Private Network Authentication Protocol. *Information Network Security*(09),1-7. <http://netinfo-security.org/CN/10.3969/j.issn.1671-1122.2021.09.001>
18. Holzmann, G. J. (1997). The model checker SPIN. *IEEE Transactions on software engineering*, 23(5), 279-295. <https://doi.org/10.1109/32.588521>
19. Maggi, P., & Sisto, R. (2002). Using SPIN to verify security properties of cryptographic protocols. In *Model Checking Software: 9th International SPIN Workshop Grenoble, France, April 11–13, 2002 Proceedings* 9 (pp. 187-204). Springer Berlin Heidelberg.
https://doi.org/10.1007/3-540-46017-9_14
20. Chen, S., Fu, H., & Miao, H. (2016, June). Formal verification of security protocols using Spin. In *2016 IEEE/ACIS 15th international conference on computer and information science (ICIS)* (pp. 1-6). IEEE. <https://doi.org/10.1109/ICIS.2016.7550830>
21. Xiao, M., Song, W., Yang, K., OuYang, R., & Zhao, H. (2022). Formal Analysis of the Security Protocol with Timestamp Using SPIN. *Computational Intelligence and Neuroscience*, 2022. <https://doi.org/10.1155/2022/2420590>
22. Edelkamp, S., Leue, S., & Lluch-Lafuente, A. (2004). Directed explicit-state model checking in the validation of communication protocols. *International journal on software tools for technology transfer*, 5, 247-267. <https://doi.org/10.1007/s10009-002-0104-3>
23. Ninet, T., Legay, A., Maillard, R., Traonouez, L. M., & Zendra, O. (2019, August). Model checking the IKEv2 protocol using Spin. In *2019 17th International Conference on Privacy, Security and Trust (PST)* (pp. 1-7). IEEE. <https://doi.org/10.1109/PST47121.2019.8949057>
24. Galaudage, S., Talbot, C., Nagar, T., Jain, D., Thrane, E., & Mandel, I. (2021). Building better spin models for merging binary black holes: evidence for nonspinning and rapidly spinning nearly aligned subpopulations. *The Astrophysical Journal Letters*, 921(1), L15.
<https://doi.org/10.3847/2041-8213/ac2f3c>
25. Liu, M. Z., Wu, T. W., Sánchez, M. S., Valderrama, M. P., Geng, L. S., & Xie, J. J. (2021). Spin-parities of the P c (4440) and P c (4457) in the one-boson-exchange model. *Physical Review D*, 103(5), 054004. <https://doi.org/10.1103/PhysRevD.103.054004>
26. Nyangaresi, V. O., Abduljabbar, Z. A., & Abduljabbar, Z. A. (2021, December). Authentication and Key Agreement Protocol for Secure Traffic Signaling in 5G Networks. In *2021 IEEE 2nd International Conference on Signal, Control and Communication (SCC)* (pp. 188-193). IEEE. <https://doi.org/10.1109/SCC53769.2021.9768338>

Qianli Wang is born on July 22, 2002, studies at Software School, East China JiaoTong University, Nanchang, China, seriously conduct scientific research and has a strict plan for the future.

Received: June 11, 2023; Accepted: September 10, 2023.

PI-BODE: Programmable Intraflow-based IoT Botnet Detection system

Dorđe D. Jovanović and Pavle V. Vuletić

School of Electrical Engineering, Bulevar kralja Aleksandra 73,
11000 Belgrade, Serbia
jd185001p@student.etf.bg.ac.rs
pavle.vuletic@ etf.bg.ac.rs

Abstract. In this paper, we propose a Programmable Intraflow-based IoT Botnet Detection (PI-BODE) system. PI-BODE is based on the detection of the Command and Control (C&C) communication between infected devices and the botmaster. This approach allows detecting malicious communication before any attacks occur. Unlike the majority of existing work, this detection method is based on the analysis of the traffic intraflow statistical parameters. Such an analysis makes the method more scalable and less hardware demanding in operation, while having a higher or equal level of detection accuracy compared to the packet capture based tools and methods. PI-BODE system leverages programmable network elements and Software Defined Networks (SDN) to extract intraflow features from flow time series in real time, while the flows are active. This procedure was verified on two datasets, whose data were gathered during the time span of more than two years: one captured by the authors of the paper and the other, IoT23.

Keywords: Botnet detection, Machine learning, IoT malware, programmable networks.

1. Introduction

Botnet is a network of computers (bots) that are under the control of a malicious hacker - botmaster. Botmasters use the devices under their control for various types of malicious activity, such as: performing Distributed Denial of Service (DDoS) attacks, spreading ransomware, stealing personal information, unwanted digital currency mining, and other [1]. Botnets came into the spotlight with the devastating attacks of the Mirai botnet which at one moment consisted of more than 600.000 devices. Although the peak of the first Mirai infection was in 2016, malware based on the Mirai code still exists and is active in creating new botnet infrastructures. For example, there has been an emergence of new botnets built using a variant of Mirai with the addition of recent exploits in the networking equipment [2]. Furthermore, new botnet generating malware appears every day, often reusing the code of the previous malware, making small changes (e.g., changing IP addresses or some strings in the messages exchanged) in order to avoid the detection or even mixing the features of multiple malware families. The recently discovered Dark Nexus botnet malware which is built on top of Mirai and QBot code [3] is one such case. For many recent botnet malware samples it is difficult

to tell which family they belong to, and multiple tags in relevant malware databases like URLhaus [32] are assigned to them.

In the botnet infrastructure, the botmaster communicates with the infected computers via the command and control (C&C) channel. One use of the C&C channel by the botmaster is to maintain the list of active bots. Botmaster either periodically polls the bots or requires periodic messages from bots using so-called C&C heartbeat packets. Another key use of C&C is to initiate the attack or to send other commands to the bots. By exploring C&C dynamic behavior patterns and creating a system that can efficiently discover botnet communication, it is possible to stop the bots before they become activated by the botmaster and involved in an attack. Since the C&C channel is a single point of failure for the botnet and its detection and mitigation fully disables the control exerted upon the bots, C&C channels evolved in time, and their detection avoidance techniques (e.g. Domain Name System (DNS) fluxing or Domain Generation Algorithms (DGA)[4]) became more sophisticated. However, our preliminary investigation [5] of the recorded samples of IoT malware did not show sophisticated detection avoidance techniques among the IoT malware samples. It revealed similar C&C behavior among multiple malware families due to the previously mentioned code reuse. C&C heartbeat communication differed in some aspects (e.g., strings in packets, the number of packets exchanged in bursts, IP flags, heartbeat initiators), but on the other side kept some features with relatively stable and similar values (e.g., low and constant bit rates, flow symmetry, periodicity and so on) which can be used for botnet detection.

Botnet or botnet-based attack detection are nowadays based on traffic statistics analysis. In this type of analysis there is a trade-off between the richness of data on one hand and network capacity, storage and processing capabilities on the other. Flow-based detection methods (that use e.g. IPFIX or similar flow gathering mechanisms) [6] store only the digest information about each network flow upon its completion (duration, number of packets and bytes). However, these methods have several drawbacks. First, full information about the flow is recorded after each flow ends, meaning that it can provide information about the attack or malicious behavior post factum. For long network flows, such as C&C heartbeats, communications last for hours or even days before the bot is activated for the attack. Such flows are either cut into multiple separate flows at the active flow timeout, which destroys the original flow properties, or stored too late to be detected during the C&C operation. Second, the global flow statistics are too coarse-grained and do not provide enough information about the intraflow dynamics, preventing the detection of some malicious activity. The alternative to this method is the analysis based on full packet capture, which provides an insight into all the packets on a specific link. The drawback to this is large overhead in processing of such data. One day of traffic recorded on a 100Gbps link amounts to approximately 1 petabyte of data that needs to be stored and processed. Furthermore, with most of the internet traffic being encrypted nowadays, stored packet data cannot provide any information apart from the size of the payload, rendering this approach highly inefficient. Therefore, in this paper we propose a PI-BODE system that lies in between the two aforementioned approaches. The system, based on Software Defined Networking principles, gathers flow statistics, enriched with the set of intraflow statistical parameters, while the flows are still active. Botnet detection is based on detecting C&C communication patterns through the analysis of intraflow statistical features, which to the best of our knowledge were not used before.

The solution was designed and evaluated using the analysis of real malware captures which were gathered in the Laboratory for information security at the School of Electrical Engineering in the period between June 2019 and October 2020 (ETF-IoTB dataset), and the IoT23 dataset captured in the period 2018-2020. The experimental results show that the PI-BODE is capable of achieving the same or higher level of detection accuracy as in the similar systems that use full traffic analysis, while it requires up to two orders of magnitude less storage space.

The structure of the paper is as follows. Section 2 provides an overview of the existing research literature. Section 3 discusses the datasets that are often used in botnet detection research, their properties and presents in detail the datasets that were used in this research. Section 4 describes the methodology of gathering and choosing the intraflow statistical features analysis. It also describes the proposed PI-BODE, a Software Defined Networking (SDN)-based intraflow statistics capturing system. Section 5 gives an overview of the data science pipeline used in this research. It describes each classifier used, feature selection and estimator parameter tuning steps. In Section 6, we presented the scores of the PI-BODE machine learning for both datasets. Obtained results are compared to the results from the related research that uses C&C communication detection. Finally, Section 7 summarizes the paper.

2. Related Work

Devastating botnet attacks on the computing and communication infrastructure provoked an increased research interest in their detection and mitigation in the last decade. Two most common botnet-related research topics are: 1. detecting the attacks and attack patterns (most often DDoS), which presents the larger part of the research work, and 2. detecting the botnet infrastructure and bot behavior. Since DDoS attacks are most often characterized by a high number of connections from various IP addresses or a high total volume of traffic, the former is done by finding anomalies in the numbers of connections and total traffic volume from the packet or flow captures [6]. However, such an approach provides information about the attack after the damage was made. For our research on C&C communication detection, more relevant is the second group of related research where we focus on the most recent papers. A thorough overview of these, mainly machine-learning based approaches is given in [7]. The key differences among the previous studies were the set of traffic features that were used to detect botnet behavior, the sources of information about the traffic (packet or flow trace), the type of botnet analyzed and the dataset that was used and machine learning methods.

One of the first C&C-related studies [8] focused on Internet Relay Chat (IRC) based C&C communication by using 16 features extracted from the full packet header traces. These features included regular statistics that can be obtained through network flow analysis, as well as packet size histogram and variances of the bytes and packet inter-arrival times for each flow. On the other side, a more recent study [9] provided an enriched traffic model and a dataset with 29 recorded and 14 generated traffic features for specific groups of flows and 3 category fields. Meanwhile, some authors explored traffic properties which cannot be found in the previously mentioned datasets, such as the periodicity in botnet communication for HTTP-based C&C communication [10, 11]. Wang et al. [12] created a BotMark system, which combines graph-based and statistical

features of traffic and hybrid analysis using similarity, stability and anomaly scores. This method uses the aforementioned scores as input for the ensemble method to classify network traffic. One of the particularly interesting features of the BotMark system is the detection method, which relies on the observed flow similarity (multiple bots have similar communication patterns with the botmaster) as the basis for the botnet communication detection.

Cusack et al., created a ransomware detection system based on machine learning and SDN-based feature extraction [13]. The network feature set in [13] includes features such as the interarrival times, the ratio between the inflow and outflow packet counts and burst lengths, which have specific values in the analyzed ransomware communication. Another paper that proposed the usage of SDN for detecting threats on the Internet of Medical Things is that of Liaqat et al. [14]. Authors used SDN to collect packet data and make forwarding decisions, which can be leveraged to stop botnet spreading. The detection mechanism uses Convolutional Neural Networks and Cuda Deep Neural Network Long Short-Term Memory (LSTM). Paper by Bilge et al. [15] focuses on the detection of botnet C&C communication using NetFlow statistics as parameters for the machine learning algorithm. Authors noticed some characteristic patterns of the C&C communication in the set of malware samples they analyzed (e.g., periodic generation of new flows, flow sizes, client access patterns, and inter-flow temporal behavior) and used inter-flow statistics to get a richer set of features that is used for the C&C channel detection. The innovative idea presented in this paper is observing flow sizes between two endpoints over a significant period of time, with the supposition that the sizes of the flows in a botnet communication will not change significantly as time passes. The main difference between [15] and our paper is that in [15] authors calculated statistics based on completed flows between two endpoints at a given time, while our paper adds fine-grain statistics of each individual flow. Blaise et al. in [16] propose an anomaly detection technique that spots the changes in the usage of a single port to identify botnets. The method is oriented towards the initial phases of the infection - TCP scans, and is adapted to detect even slow and stealthy changes. Koroniotis et al. worked on the detection of various attacks towards the IoT infrastructure including probing attacks (scanning and fingerprinting), DoS and information theft [9]. The paper by De la Torre Parra et al. [17] provides an overview of IoT based botnets and their detection methods. Authors used LSTM neural networks for the attack detection. The attack set in that paper included various types of traffic floods, but also scanning as a preparatory attack phase and sending spam data. Kurniabudi et al. in [18] analyzed feature selection using information gain as a criterion and created several classification machine learning models. It was shown that the model prediction depends on the number of features selected, and that the optimal number of those features differs from model to model. One significant branch of work was directed towards DNS-based botnet evasion techniques detection, such as DGA and DNS-fluxing [4,19-22]. Since the analyzed IoT malware samples did not use such mechanisms in this paper, DNS-based evasion techniques were not considered.

In contrast to previously mentioned approaches, all based on full packet capture analysis, PI-BODE uses flow information enriched with an original set of intraflow features, which creates a new type of dataset for the C&C channel detection, different from the datasets mentioned in this section. The produced dataset is significantly smaller, allowing cheaper storage and processing capabilities while providing at least the same level of detection accuracy. Flow statistics are extracted leveraging the

capabilities of the programmable network elements while the flows are still active, which enables online botnet detection. Similarly to the other research, PI-BODE uses machine learning techniques for detecting botnet C&C heartbeat communication channels.

3. Malware Samples and Datasets in Recent Botnet Research

One of the critical decisions in malware and intrusion detection research is the choice of the malware samples and dataset which are used for algorithm training and evaluation. A recent study [23] emphasized some issues with the datasets that are often used for the research. Some of those issues are dataset age or conditions under which the traffic was recorded, which become irrelevant due to the changes in the attack patterns over the years. Such an example is the NSL-KDD dataset, which was recorded more than twenty years ago, yet it is still being used in some recent research studies [24-26]. The use of this dataset can be justified for research on the volumetric type of (D)DoS attacks, which do not change a lot in nature over time. However, other botnet features, like other information security threats, tend to constantly change their pattern of behavior to avoid detection.

Some features, such as scanning, malware proliferation mechanisms or C&C communication continuously change in time [27]. It is a well-known fact that Mirai malware had 24 versions only in the first two months since it appeared [28], and its derivatives still appear daily. For such changing features, the use of outdated datasets gives results of limited usability. Besides, the age of the datasets, Al-Hadrami et al. [23] highlighted the problems with labeling these datasets. For instance, the information about the conditions under which they have been recorded. However, we would like to emphasize another issue with recent datasets that are commonly being used for this type of research. Datasets are recorded over a limited time presenting a snapshot of the threat landscape at that brief moment. Table 1 summarizes some of the recently used datasets for botnet detection.

Table 1. Dates when some of the commonly used datasets were recorded

Dataset	Used in	Year of recording	Recording duration
AWID	[26]	2015	9 days
ISOT	[8]	2017	7 days
CIC-IDS	[18][24][26]	2017	5 days
BotMark	[13]	2016	16 days
Bot-IoT	[8][10]	2018	6 days in a one-month period
NIMS	[29]	2014	1 day

To avoid models overfitted to one specific dataset and threat, we argue that the dataset should be created over a longer period and continuously refreshed. Also, the

research should focus on finding the features which rarely change, while the verification of machine learning models must be done on different, most recent datasets, captured over longer periods. Therefore, we decided to create our own dataset (ETF-IoTB), based on recorded malware samples over a period of more than one year, while performing verification with the IoT23 dataset [30]. The dataset details are given in the remainder of this section.

3.1. **ETF-IoTB Dataset**

The ETF-IoTB dataset was obtained by recording the traffic of the devices infected with the malware samples [31]. Malware samples were downloaded from the links in the URLHaus [32] database of the recent malware distribution points. Malware was executed on RaspberryPi 3 devices with Raspbian OS. RaspberryPi devices were connected to the internet without any protocol filtering, apart from protecting the local network infrastructure from malware lateral movement. The created dataset fulfills several recommendations of what constitutes a correct botnet dataset [33]: includes real botnet communication and not simulation, has unknown/regular traffic, has ground-truth labels for training and evaluating the methods and includes different types of botnet malware. Benign traffic was traffic from a regular workstation during the day.

The infected devices were constantly monitored in order to detect the situations when there is a sharp increase of the traffic volume as an indication that the infected machine is a part of the DDoS attack. The devices typically worked in the pre-attack phase, when their initial and C&C heartbeat communication were recorded, and in few cases were stopped as soon as the DoS attacks from the infected device were noticed, as described in the provided dataset. Malware dynamic behavior samples were downloaded in the period between June 15th 2019 and October 15th 2020. Two IoT malware families which are the most common today were analyzed: Mirai and Gafgyt. The dataset consists of sixteen Mirai, eighteen Gafgyt and one NanoCore sample. More details about the dataset and noticed patterns of C&C communication, as well as initiating attack commands are given in our previous work [5].

3.2. **IoT23 Dataset**

The IoT23 dataset is the latest dataset provided by the Technical University in Prague. Dataset consists of labeled benign and botnet traffic collected on IoT devices. The IoT23 dataset contains 23 scenarios, 20 of which contain malware traffic, while the remaining 3 contain benign traffic. Traffic samples in this dataset were captured in the period between 2018 and 2020. Malware was also recorded on RaspberryPi devices, while the benign traffic scenarios were recorded on various IoT devices: a Philips HUE smart LED lamp, an Amazon Echo home intelligent personal assistant and a Somfy smart door lock. The IoT23 dataset consists of 7 Mirai, 1 Muhstik, 1 Kenjiro, 2 Torii, 1 IRCBot, and 1 Gafgyt sample. The datasets are labeled and contain 8 labels related to the C&C communication.

4. Intraflow Statistical Feature Analysis

In this section we present PI-BODE principles of operation and gathering network intraflow statistical features.

4.1. Capturing Intraflow Information

Intraflow statistical parameters can be obtained from the packet capture (pcap) files using a software tool similar to Joy [34], which is capable of extracting some intraflow features (e.g., per-flow packet size probability distribution, entropy or Walsh-Hadamard transform).

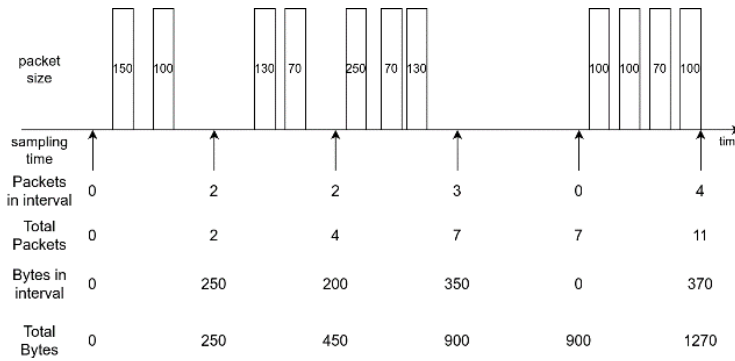


Fig. 1. Sampling flow statistics - A visualization on how packet size in bytes and number of packets is collected from a time series

Unlike this approach, PI-BODE leverages the SDN OpenFlow features for real-time statistics capturing. Instead of capturing each packet, our system samples the flow statistics periodically and gathers the flow metrics at each sample time. Figure 1 shows how flow data is being captured from the SDN switch. By sending the OFPFLOWStatsRequest message periodically, the controller requests from the switch the total number of packets and the total amount of data for each flow. By sampling these counters and calculating the difference between the results of the two consecutive samples, per-flow time series for packet and byte counts are created. They are then used as a basis for various statistical features calculations, described in Section 4.3.

Two configuration parameters are: sampling period and flow idle timeout value. For the first parameter we used 1 second period as a rule of thumb, which showed reliable detection results and low network overhead. The choice of the second parameter is important for those flows, which have either long idle periods or do not have explicit end of flow signalization (e.g., UDP flows). Longer idle timeout means more overhead because expired flows will remain longer in the switch tables, but also simpler processing at the analysis station. Processing is simpler because there is no need to concatenate the parts of the flows with long idle periods if they are broken in multiple flows. We used 2 minutes for the idle timeout, as the longest periods in the C&C heartbeat communication we observed were 60s, thus ensuring non-interruptible C&C

flow recording. These parameters also define the key limitations of the PI-BODE detection system. Worse detection results can be expected if the C&C communication patterns change in a way which would make them hardly noticeable with the chosen set of configuration parameters (sampling period and idle timeout). Botnet heartbeats with a period shorter than sampling period or longer than the idle timeout would be difficult to detect. However, in case of such changes, the system can be easily reconfigured to adapt to the changes.

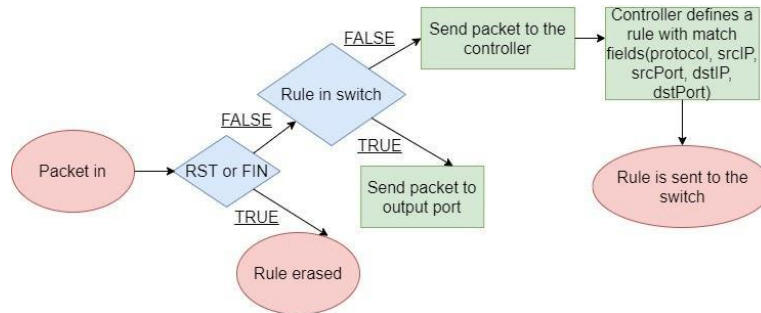


Fig. 2. Flow statistics gathering algorithm - Algorithm used by the SDN controller to perform flow switching

The simple algorithm for the TCP flow statistics gathering is given in Figure 2. For each TCP flow, the key events (first packet and terminating handshake) cause flow rules to be added or deleted in the switch. The SDN switch works as a Layer-2 transparent bridge where packets are being forwarded per flow. Each new flow, defined as a tuple (Source IP, Destination IP, Source Port, Destination Port, Transport protocol) creates a new entry in the controller's flow statistics time series database. Each flow has an OFPFF_SEND_FLOW_REM flag set to force the switch to send the summary information to the controller upon the flow removal. UDP or TCP flows with idle periods longer than the idle timeout are removed from the flow time series database upon the expiration of the idle timeout values.

Intraflow data series provide a more compact dataset and a much smaller network overhead. The exact gain in the amount of transferred traffic from the network element needed to do the attack detection is difficult to estimate, as it depends on various parameters in a given network traffic sample like: flow length distribution, number of packets per flow, flow ending methodology and others. We will illustrate the difference using one example which is taken from our packet capture [31]. One of our packet capture files with the non-malware traffic contained a total of 1,474,536 packets, which created 5,171 network flows of a total size of 2.496 gigabytes. The duration of the packet capture was 12,641s with 116 packets or 196 kilobytes per second on average transferred from the network element towards the analysis station. That same traffic when we used flow parameters sampling rate of 1s created the average traffic from the network element towards the controller in the range between 685 and 6,979 bytes per second for varying idle flow timeout values between 5s and 600s respectively that is a reduction in storage and network overhead between 28 and 280 times.

4.2. SDN Based Botnet Detection System based on intraflow statistics

Since most of the IoT devices do not possess sufficient hardware capacities, software on them is often simple and prone to attacks. Adding additional security measures on the IoT devices is not always possible, meaning security measures must be implemented on the network level. The optimal solution is to place the attack detecting device in the line of the packets that flow between the protected devices and the internet and act as both intrusion detection and prevention systems (IDS and IPS) (Figure 3).

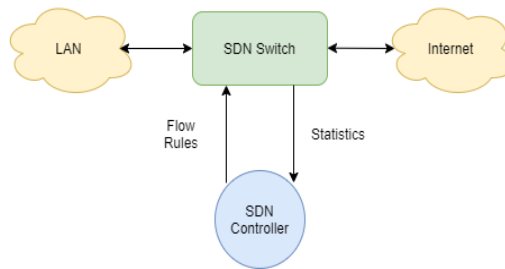


Fig. 3. IDS system using SDN - A schematic diagram of a system that protects the LAN from botnet attacks

The PI-BODE SDN controller software consists of three components (depicted in Figure 4.): Flow engine, Data collection thread and Analysis thread. The flow engine enables packet forwarding through the network. Each network flow has its own entry in the switch's flow table, but in terms of packet rewriting the switch does the basic Layer-2 forwarding. The data collection thread is used to collect packet and byte counts for all flows each second, and the collected data are then put into their appropriate time series as described in Section 4.1.

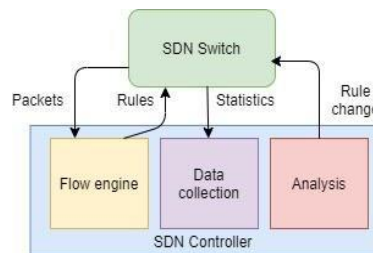


Fig. 4. SDN controller’s internal structure - Principal components of the SDN controller, as well as its interactions with the switch

For all flows we also captured flow duration, total number of packets and number of entries in the empirical distribution. These parameters are not deemed flow features that are used in the detection process but are used to calculate the other features. Furthermore, flow duration is also the criterion that qualifies the flow to be used in the detection process. All flows shorter than 60s are not used for the botnet C&C detection. Some bots within the ETF-IoTB dataset (for example, xs.arm7 and nuclear.arm7) that

use periodic C&C pings and each ping represents a different flow (uses different port numbers for each ping). In these cases, defining a flow as double (src_IP, dst_IP) can capture the C&C communication and make flow tables more compact. However, in the remainder of the document we worked with tuples as defined in Section 4.1.

4.3. Intraflow Features

As described in Section 4.1, for each flow, packet and byte count samples are put into two time series, with a 1s time bin. From those time series we extracted 22 statistical features. Table 2 lists 14 features that will be analyzed in the remainder of this paper, their brief descriptions and Area Under Curve (AUC) values. The detailed discussion about the use of AUC values is given in Section 5.2.

Features 1-11 are extracted from the two time series. Features 12-14 are extracted from the empirical distribution of the flow quiet times. The flow quiet time is the number of contiguous time bins whose byte count value is 0. Empirical distribution of the flow quiet times consists of the number of occurrences of different flow quiet times in a bytes flow time series. The reason for creating such a distribution is that the observed botnet C&C behavior consists of periodic exchanges between the C&C and bot for most of the time. Therefore, by capturing intervals of flow inactivity and analyzing their distribution, it is possible to detect this periodic behavior. The explanation of the flow collection process was given in Section 4.2.

5. C&C Detection Methodology

To detect IoT based C&C communication, a regular machine learning method is applied and further optimized using a data science pipeline which comprises the following steps: data extraction and handling, feature selection, hyperparameter tuning and classification. One step that is often used as a part of the data science pipeline is data transformation, namely, normalization and standardization [36]. These procedures transform range or data distribution for each feature, which allows certain classifiers to show better performance. However, there is a reason for not using data transformation in machine learning models that are applied to security systems. Normalization and standardization are based on transforming feature values in a certain dataset, using statistical properties of each feature. The goal of the detection system is to detect malware unknown during the training phase. With transformed data, new malware can have features whose values are out of bounds of the normalized or standardized dataset that was used for training.

Table 2. Intraflow features used in the C&C flow detection

No	Feature	Feature description	AUC
1	Average flow throughput (bytes/s)	Number of bytes per second of the time series which is re-calculated after each new sample is added to the time series.	0.96
2	Average number of packets in a flow (packets/s)	Number of packets per second of the time series which is re-calculated after each new sample is added to the time series.	0.93
3	Median of the throughput time series (median)	Median value of all time bins in throughput time series	0.553
4	Average throughput difference	Difference between the average flow throughputs for bidirectional sessions	0.738
5	Average throughput ratio (Bidirectional bytes ratio)	Ratio between the average flow throughputs for bidirectional sessions.	0.805
6	Standard deviation (Std)	Throughput time series standard deviation	0.863
7	Correlation (Corr)	Pearson correlation coefficient of the throughput time series	0.551
8	ADF Stat value	Augmented Dickey–Fuller test (ADF test): a statistical test which determines whether a time series is stationary or not.	0.511
9	ADF Test p-value	p-value for the ADF test	0.607
10	Autocorrelation (Autocorr)	Autocorrelation of the bytes count time series calculated at the lag equal to the highest probability in the empirical distribution of the flow inactive periods (described below). This feature aims to capture the periodic behavior of the flow.	0.985
11	Entropy	Throughput time series entropy calculated using the formula $H(X)=-\sum P(x_i)\ln(P(x_i))$, where $P(x_i)$ is the probability of a packet having given size.	0.767
12	Best/Other ratio	The ratio of the probabilities of the value that appears the most often in the empirical probability distribution and the most probable highest value in the periodicity dictionary to the sum of other values	0.893
13	Best/Second ratio	The ratio of two most probable values in the empirical distribution	0.85
14	Top Two	Sum of top two values that appear the most often in the empirical distribution	0.849

5.1. Classifier Description

The classifiers used in this paper are: K-nearest neighbor (KNN), logistic regression, and random forest from the Python sklearn library. The K-nearest neighbor algorithm is a non-parametric method used in pattern recognition [35]. Each data point is represented

by a n -dimensional vector in feature space. The training phase of the algorithm consists of storing the data points with labels. The classification phase begins by considering each new data point's position in feature space. Then, by using a distance metric, k -nearest neighbors are determined, and a data point is classified by majority voting.

Logistic regression uses a sigmoid function to perform classification. This machine learning model is the discrete version of linear regression. Using the data point representation as n -dimensional vectors designated with x_i , polynomial real coefficients designated with β_i , and error value designated with ε , the input to the sigmoid function has the following form:

$$f(x) = \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \varepsilon. \quad (1)$$

The value of probability is between 0 (data point certainly belongs to class "0") and 1 (data point certainly belongs to class "1"). The classes named "0" and "1" are defined before classification. During the training phase, each data point is fed into the sigmoid function, fitting the coefficients and error value so that the best fit is achieved.

Random forest belongs to a class of data science models called ensemble methods. In essence, an ensemble method combines many simple or weak classifiers, and the classification is performed using majority voting. The core classifier in this method is the decision tree, consisting of nodes and leaves. Each node contains a boundary value for a feature, whilst the leaves contain the information to which class a certain data point belongs. Selecting a subset of features, as well as a ranking function, the features are ranked and the decision tree is constructed. This is done by putting the features as boundary values inside the decision tree, going from the topmost ranking feature down to the lowest. The random forest consists of a certain number of random trees, and the class for the data point is determined by majority voting.

5.2. Feature Selection

Dimensionality reduction is an important step in order to create a classifier suitable for the implementation in the network elements in terms of memory and processing power consumption. For this purpose, the area under the curve (AUC) for receiver operating characteristic (ROC) calculated as if each feature alone was used for the classification is used as a measure of the impact that the specific feature has on the classification.

The ROC is a plot that shows for each value of a feature the ratio of true positive rate (TPR) to true negative rate (TNR). The number of true positives and the number of true negatives is designated as TP and TN, while the number of false positives and the number of false negatives is designated as FP and FN. TPR and TNR two values are calculated as follows:

$$\text{TPR} = \text{TP} / (\text{TP} + \text{FN}), \text{ TNR} = 1 - \text{Specificity} = 1 - \text{TN} / (\text{TN} + \text{FP}). \quad (2)$$

The ROC is a probability curve, and the AUC value shows how well a certain feature separates the data correctly into classes. The AUC holds values in the [0,1] interval, where a value closer to one indicates a better feature score. The features with the AUC values higher or equal to 0.85 are autocorrelation, throughput (bytes/s), packets/s, Best/Other ratio, Standard deviation, Best/Second ratio. However, the features that are chosen for the classification process should not be mutually correlated as this means that they describe the same or similar dataset property. Therefore, a correlation matrix

was used to test the correlation between all pairs of features. The correlation matrix is shown in Figure 5. High correlation (greater than 0.8) is observed between the bytes/s and packets/s features. Therefore, the five features that are chosen as input features to the classification process are Autocorrelation, Throughput (bytes/s), Best/Other ratio, Standard deviation of the throughput time series and Best/Second ratio.

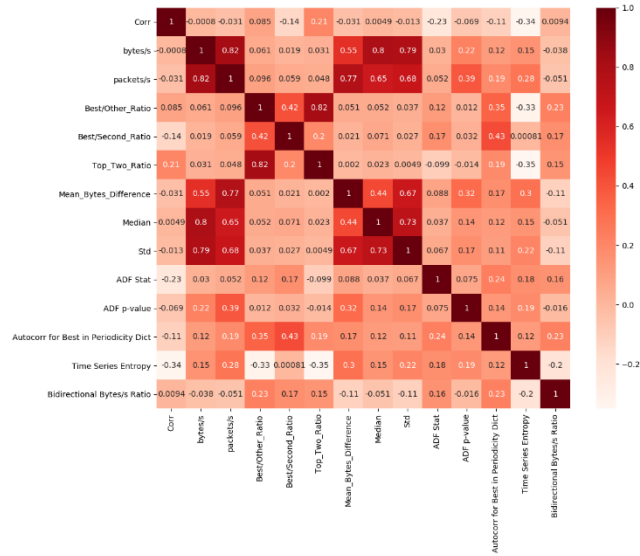


Fig. 5. Correlation matrix for feature selection - A correlation matrix of all the input features, which is used for feature selection.

5.3. Estimator hyperparameters

Each estimator has a set of hyperparameters, which define its behavior. In this phase, we analyzed the combination of estimator parameters that maximizes the scores used in classification using a “fit” and a “score” method implemented by the GridSearchCV method [37] in the sklearn Python library. The method is slightly modified as described below to fit the analyzed datasets which are disbalanced. The set of scores used in this paper are precision (PREC), recall (REC), and f1-score (F1) [35]. These scores are given in the equations below:

$$PREC = TP / (TP + FP), \tag{3}$$

$$REC = TP / (TP + FN),$$

$$F1 = 2 \cdot PREC \cdot REC / (PREC + REC).$$

Precision is a measure which shows how many false negatives are there during classification, while recall is a measure which shows how many false negatives are there

during classification. The F1 score represents a measure whose objective is to analyze the trade-offs between correctness and coverage in classifying positive instances [38], as it includes both precision and recall. Although accuracy is a more intuitive metric, these three measures are chosen because there is a difference in sizes of normal and botnet flows that is larger than 20 percent, which is considered a disbalanced dataset. In case of disbalanced datasets accuracy can be biased towards the majority class. Even high accuracy values can yield poor classification rates on minority classes and classifiers can have low predictive power [39]. Botnet C&C datasets are disbalanced because botnet C&C communication is a relatively rare occurrence in network traffic compared to all the other traffic and produces typically one or few flows at a time.

Balanced sampling was used to split data into training and testing subsets with a 90/10 ratio. The data is split in such a manner, in order to preserve the ratio of classes from the dataset, which is known as stratified sampling. A similar approach was presented in [40], in order to account for class imbalance in the dataset. After splitting the dataset, the model is trained and tested, and the scores are extracted. This procedure is repeated 100 times, and mean values are calculated for all scores (Monte Carlo cross-validation [41]). This approach was taken by the authors in order to have as much variation to the training and testing set scenarios, while keeping the number of scenarios tested to a minimum. Again, because the dataset is disbalanced, this method is more appropriate than the ubiquitous k-fold cross-validation, which splits the whole dataset into k equal parts. Such an approach produces some parts of the dataset to have low presence of botnet data points. After calculating all the scores, the list of parameter combinations is searched for those having the maximum values for all scores. In the following paragraph, an overview of the tested hyperparameters for both datasets is given. After the name of each hyperparameter, the best value is given in parentheses. If there is one value, that means that the value is best for both datasets, while if there are two values, the first one refers to the ETF-IoTB dataset, and the other refers to the IOT23 dataset.

The hyperparameters that were considered for the KNN classifier are the following: number of neighbors (5), weight function (uniform), neighbor computation algorithm (ball tree), and the distance metric (Manhattan). The hyperparameters that were considered for the logistic regression classifier are the following: inverse of the regularization strength (0.25), class weights (none), option of adding a constant to the decision function (false), maximum number of iterations (100), solver (newton-cg, lbfgs), and the tolerance for the stopping criteria (0.001, 10^{-5}). The hyperparameters that were considered for the random forest classifier are the following: class weights (balanced subsample), ranking function (entropy), max number of features (none), and the total number of estimators (200).

6. PI-BODE Classification Evaluation

This section presents and discusses the results of machine learning classification for three estimators: KNN, Logistic regression and Random Forest, using the hyperparameter values determined in the previous section. Precision, Recall and F1 scores for the PI-BODE detection method are given in Tables 3 and 4 for the ETF-IoTB and IoT23 datasets, respectively.

Table 3. Scores for all the estimators for the ETF-IoTB dataset

Model	botnet precision	botnet recall	botnet f1 score	normal precision	normal recall	normal f1 score
KNN	0.9363	0.9141	0.9218	0.9967	0.9973	0.9970
Logistic regression	0.9131	0.9247	0.9150	0.9971	0.9962	0.9967
Random forest	0.5954	0.6240	0.5994	0.9859	0.9948	0.9902

Table 4. Scores for all the estimators for the IOT23 dataset

Model	botnet precision	botnet recall	botnet f1 score	normal precision	normal recall	normal f1 score
KNN	0.8683	0.8759	0.8690	0.8558	0.8382	0.8423
Logistic regression	0.8347	0.8778	0.8510	0.8522	0.7858	0.8093
Random forest	0.5564	0.6071	0.5780	0.7566	0.8733	0.7724

Several conclusions can be drawn from these results. First, the highest scores on both datasets prove that this method can be used to detect botnet C&C communication with high precision. Second, even though the feature selection was done on the ETF-IoTB dataset, the results of the detection method on the IoT23 dataset that contains a different set of malware samples (Muhstik, Kenjiro, Tori in addition to those in the ETF-IoTB dataset) revealed similar precision compared to the other research papers. It can be concluded that the key C&C features in different malware samples are not largely different in two datasets. Also training a model, which uses features obtained from the intraflow statistics, can detect the C&C communication of the malware samples that are not in the dataset used to train the model, so the zero-day malware detection is possible. Third, the results show that for both datasets, KNN and Logistic Regression classifiers have given the best results, while having little discrepancies between the precision and recall scores. This means that there is little difference between the number of false negatives and false positives, demonstrating detection stability. The consistency in classifier performance shows that the proposed method performs well regardless of the dataset used. On one hand, KNN classifier shows a slightly better score than Logistic Regression classifier, while on the other KNN has much greater memory usage and is impractical to be used in network elements. Since ultimately these models should be implemented in a security system, using a Logistic Regression classifier is recommended, because it gives nearly the best scores among all the classifiers, while keeping the memory usage at a lower level.

Papers in references [11] to [13] have, similar to our approach, sought to detect botnets and ransomware via the C&C communication. In paper [11], the research showed an accuracy of 80% in classifying botnet periodicity, which is lower than scores obtained in our research for the best classifiers. Tables 5 and 6 show research results presented in papers [12] and [13], respectively. In paper [12], the BotMark detection system has shown a high accuracy (0.9994) and low false positive rate scores. However, since the F1-scores are low for all previous research methods (0.115 for BotMark, compared to 0.915 and 0.85 for PI-BODE Logistic Regression) while the false positive rate is also low, this means that the false negative rate is high, which further implies that the stability of the BotMark classifiers is low, and certainly much lower than in PI-BODE.

Table 5. Scores for all the methods in paper [12]

Method	F-score	Accuracy	True Positive Ratio	False Positive Ratio
Similarity	0.087247	0.9904	0.9836	0.009645
Stability	0.060732	0.9868	0.9115	0.013181
C-flow	0.152788	0.9949	0.9836	0.005108
Graph	0.034811	0.9166	0.9836	0.083478
BotMark	0.115207	0.9994	0.9836	0.000641

Table 6. Scores for all the methods in paper [13]

No of features	Precision	Recall	F1-score
28	0.83	0.89	0.86
8	0.86	0.87	0.87

Authors in [13] use precision, recall and F1-score as scores for their classifier. In both cases presented in [13], with 8 and 28 features, score values are in the range 0.83 to 0.89. This is worse classification performance than with the PI-BODE whose scores when the system is trained, verified and tested on the ETF-IoT dataset, are in the range 0.914 to 0.936 for the KNN and Logistic Regression classifiers. Even with the classifier training on the ETF-IoT dataset and testing on the IOT23 dataset (zero-day detection case), the scores are in the range 0.83 to 0.88 which is comparable to the scores obtained in [13]. Also, what needs to be taken into account is the fact that both of these papers show only the scores in regards to the whole dataset. In the case of balanced datasets, scores can be presented this way. However, since malware represents a minor part of each network traffic, such datasets are disbalanced by nature. Scores presented in this way should be treated with caution, since there is no information on the machine learning model's performance on both normal and malware traffic. At last, but not at

least ransomware C&C communication detection described in [13] is based on full packet capture analysis meaning a much higher network and storage overhead.

7. Conclusion

In this paper, we proposed and demonstrated a novel approach for the detection of botnet C&C communication based on SDN and intraflow statistics. It provides more traffic description features than flow-based methods, while not using the analysis of all packets on a link. Thus, this method represents an in-between solution that saves storage and processing power up to two orders of magnitude, yet provides enough detail about network conversations for distinguishing malware and normal traffic. Experimental results have shown that PI-BODE achieves the same or higher level of botnet detection accuracy as in the similar systems which use full traffic analysis. Other conclusions can be drawn from the presented research. First, the results prove that botnet C&C communication can be detected using the intraflow statistics and proposed traffic features. Second, since PI-BODE training and evaluation have been done on the datasets that contain various malware samples, it shows that different versions of botnet malware have similar C&C communication characteristics and that new malware samples can be detected with the system trained on the older malware samples. Third, using an SDN controller as an intraflow statistics gathering tool allows creating a simple intrusion prevention system for the devices that lack security, such as IoT devices. Using the PI-BODE approach as a basis, future research should attempt to explore additional statistical parameters of the network flow, analyze other malware dynamic properties that can be used for the detection, explore the use of other programmable platforms (e.g., P4-based) and to assess how to improve the malware detection speed. The research presented in this paper can be extended to using other state-of-the-art machine learning algorithms, such as boosting models (such as extreme gradient-boosting and light gradient-boosting machine) and deep learning models. Also, since computer viruses are constantly evolving, training a dataset on current botnets may not be applicable to detecting botnets in the future. Therefore, future research should be directed towards developing a machine learning model with on-line learning capabilities.

References

1. Vormayr, G., Zseby, T., Fabini, J.: Botnet Communication Patterns, *IEEE Commun. Surv. Tutorials*, vol. 19, no. 4, pp. 2768–2796, 10.1109/COMST.2017.2749442 (2017)
2. Chen, R., Niu, W., Zhang, X., Zhuo, Z., Lv, F.: An Effective Conversation-Based Botnet Detection Method, *Math. Probl. Eng.*, vol. 2017, pp. 1–9, 10.1155/2017/4934082 (2017)
3. B. Krebs, Zyxel Flaw Powers New Mirai IoT Botnet Strain, Krebs on Security website, <https://krebsonsecurity.com/2020/03/zyxel-flaw-powers-new-mirai-iot-botnet-strain/> (accessed on December 21st 2022)

4. Štampar, M., Fertalj, K.: Applied Machine Learning in Recognition of DGA Domain Names, *Computer Science and Information Systems*, vol. 19, No. 1, 205-227., 10.2298/CSIS210104046S (2022)
5. Jovanović Đ., Vuletić P.: Analysis and Characterization of IoT Malware Command and Control Communication, *Telfor Journal Vol.12 No.2*, p. 80-85, 10.5937/telfor2002074B (2020)
6. Ibrahim, J., Gajin, S.: Entropy-based Network Traffic Anomaly Classification Method Resilient to Deception. *Computer Science and Information Systems*, Vol. 19, No. 1, 87-116., 10.2298/CSIS201229045I (2022)
7. Asadi, M., Jabraeil Jamali, M. A., Parsa, S., Majidnezhad, V.: Detecting botnet by using particle swarm optimization algorithm based on voting system. *Future Generation Computer Systems*, vol. 107, 95–111., 10.1016/j.future.2020.01.055 (2020)
8. Livadas C., Walsh, R., Lapsley, D.E., Strayer, W.T.: Using machine learning techniques to identify botnet traffic, *LCN*, pp. 967–974., 10.1109/LCN.2006.322210 (2006)
9. Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B.: Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset, *Future Generation Computer Systems*, vol. 100, p. 779-796, 10.1016/j.future.2019.05.041 (2019)
10. Lee, J.-S., Jeong, H., Park, J.-H., Kim, M., Noh, B.-N.: The activity analysis of malicious http-based botnets using degree of periodic repeatability, 2008 *International Conference on Security Technology*, pp. 83–86., 10.1109/SecTech.2008.52 (2008)
11. Eslahi, M., Rohmad, M. S., Nilsaz, H., Naseri, M. V., Tahir, N. M., Hashim, H.: Periodicity classification of HTTP traffic to detect HTTP Botnets, 2015 *IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, p. 119–123. 10.1109/ISCAIE.2015.7298339 (2015)
12. Wang, W., Shang, Y., He, Y., Li, Y., Liu, J.: BotMark: Automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors, *Information Sciences*, Vol. 511, p. 284–296. 10.1016/j.ins.2019.09.024 (2020)
13. Cusack, G., Michel, O., Keller, E.: Machine Learning-Based Detection of Ransomware Using SDN, In *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization - SDN-NFV Sec'18*, pp. 1–6., 10.1145/3180465.3180467 (2018)
14. Shahzana Liaqat, S., et al.: SDN orchestration to combat evolving cyber threats in Internet of Medical Things (IoMT), *Computer Communications*, Volume 160, p. 697-705, 10.1016/j.comcom.2020.07.006 (2020)
15. Bilge, L. et al.: DISCLOSURE: Detecting Botnet Command and Control Servers Through Large-Scale NetFlow Analysis, *ACSAC '12*, 10.1145/2420950.2420969 (2012)
16. Blaise, A., Bouet, M., Conan, V., Secci, S.: Detection of zero-day attacks: An unsupervised port-based approach, *Computer Networks*, vol.180, 10.1016/j.comnet.2020.107391 (2020)
17. De La Torre Parra, G., Rad, P., Choo, K.-K. R., Beebe, N.: Detecting Internet of Things attacks using distributed deep learning, *Journal of Network and Computer Applications*, vol. 163, 10.1016/j.jnca.2020.102662 (2020)

18. Kurniabudi, et al.: CICIDS-2017 Dataset Feature Analysis With Information Gain for Anomaly Detection, *IEEE Access*, Volume 8, p. 132911-132921, 10.1109/ACCESS.2020.3009843 (2019)
19. Sharifnya, R., Abadi, M.: DFBotKiller: Domain-flux botnet detection based on the history of group activities and failures in DNS traffic, *Digit. Investig.*, vol. 12, pp. 15–26, 10.1016/j.diin.2014.11.001 (2015)
20. Plohmann, D., Yakdan, K., Klatt, M., Bader, J., Gerhards-Padilla, E.: A Comprehensive Measurement Study of Domain Generating Malware, Open access to the Proceedings of the 25th USENIX Security Symposium is sponsored by USENIX, pp. 1996–2014, 10.5555/3241094.3241115 (2016)
21. Tong, V., Nguyen, G.: A method for detecting DGA botnet based on semantic and cluster analysis, *ACM Int. Conf. Proceeding Ser.*, vol. 08, pp. 272–277, 10.1145/3011077.3011112 (2016)
22. Wang, T. S., Lin, H. T., Cheng, W. T., Chen, C. Y.: DBod: Clustering and detecting DGA-based botnets using DNS traffic analysis, *Computer Security*, vol. 64, pp. 1–15, 10.1016/j.cose.2016.10.001 (2017)
23. Al-Hadhrami, Y., Hussain, F. K.: Real time dataset generation framework for intrusion detection systems in IoT, *Future Generation Computer Systems*, Vol. 108, p. 414–423., 10.1016/j.future.2020.02.051 (2020)
24. de Souza, C. A., et al.: Hybrid approach to intrusion detection in fog-based IoT environments, *Computer Networks*, 180, 107417., 10.1016/j.comnet.2020.107417 (2020)
25. Hosseini, S., Zade, B. M. H.: New hybrid method for attack detection using combination of evolutionary algorithms, SVM, and ANN, *Computer Networks*, Vol. 173, 10.1016/j.comnet.2020.107168 (2020)
26. Zhou, Y., Cheng, G., Jiang, S., Dai, M.: Building an efficient intrusion detection system based on feature selection and ensemble classifier, *Computer Networks*, 174, 10.1016/j.comnet.2020.107247 (2020)
27. Gardiner, J., Cova, M., Nagaraja, S.: Command & Control: Understanding, Denying and Detecting, vol. cs.CR, no. February, 10.48550/arXiv.1408.1136 (2014)
28. Antonakakis, M. et al.: Understanding the Mirai botnet, *SEC'17*, p. 1093–1110., 10.5555/3241189.3241275 (2017)
29. Shafiq, M., et al.: Selection of effective machine learning algorithms and Bot-IoT attacks traffic identification for internet of things in smart city, *Future Generation Computer Systems*, Vol. 107, p. 433–442. [10.1016/j.future.2020.02.017](https://doi.org/10.1016/j.future.2020.02.017) (2020)
30. Parmisano, A., Garcia, S., Erquiaga, M. J.: A labeled dataset with malicious and benign IoT network traffic. *Stratosphere Laboratory*, 10.5281/zenodo.4743746 (2020)
31. Jovanovic, G., Vuletić, P.: ETF IoT Botnet Dataset, *Mendeley Data*, V1, 10.17632/nbs66kvx6n.1 (2021)
32. abuse.ch, URLHaus, a database of malware URLs, <https://urlhaus.abuse.ch/> (accessed on December 21st 2022)
33. García et al.: An Empirical Comparison of Botnet Detection Methods, *Computers & Security*, 10.1016/j.cose.2014.05.011 (2014)

34. Joy, A package for capturing and analyzing network flow data and intraflow data, for network research, forensics, and security monitoring, <https://github.com/cisco/joy>, (accessed on September 2nd 2020)
35. Skiena, S. S.: The Data Science Design Manual, Springer, 10.1007/978-3-319-55444-0 (2017)
36. Raschka, S.: About Feature Scaling, https://sebastianraschka.com/Articles/2014_about_feature_scaling.html (accessed on May 21st 2022)
37. sklearn documentation, GridSearchCV, https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, (accessed on May 21st 2022)
38. Fernández, A. et al.: Learning from Imbalanced Data Sets, Springer, 10.1007/978-3-319-98074-4 (2018)
39. Gibert, D., et al.: The rise of machine learning for detection and classification of malware: Research developments, trends and challenges, Journal of Network and Computer Applications, Vol. 153, 10.1016/j.jnca.2019.102526 (2020)
40. Apruzzese, G. et al.: Evaluating the effectiveness of Adversarial Attacks against Botnet Detectors, 2019 IEEE NCA, 978-1-7281-2522-0/19/ (2019)
41. Dubitzky, W., Granzow, M., Berrar, D.: Fundamentals of data mining in genomics and proteomics, Springer Science & Business Media, p. 178., 10.1007/978-0-387-47509-7 (2007)

Đorđe Jovanović obtained his BSc and MSc in Computer Networks from University of Belgrade, School of Electrical Engineering (ETF). Since 2019, he began his work as an assistant researcher-trainee at the Mathematical Institute of Serbian Academy of Sciences and Arts (MISANU/SASA). Since 2022, he works as an assistant researcher at MI SANU. His research interests encompass Software-Defined Networks (SDN), computer networks, botnet detection, machine learning, optimization, and metaheuristics.

Pavle Vuletić obtained his BSc, MSc and PhD in Computer Systems and Network Architecture from University of Belgrade, School of Electrical Engineering (ETF). He used to work on all positions from network engineer to the deputy director of AMRES, national research and education network where he participated in the establishment of the first national CSIRT team. He is currently an associate professor at the ETF teaching Data Security, Computer Systems and Network Security, Advanced Computer Networks and SDN courses, leads the Laboratory for Information Security at ETF. His research interests span from data protection and privacy, network and systems security, network and system performance evaluation to the programmable networks and network and systems management.

Received: February 22, 2023; Accepted: September 10, 2023.

Feature Parameters extraction and Affective Computing of Voice Message for Social Media Environment

Peng Jiang¹, Cui Guo², and Yonghui Dai^{3,*}

¹ Jingan Branch Campus, Shanghai Open University,
Shanghai 200040, China
jzhpmail@163.com

² Shanghai Lifelong Education School Credit Bank Management Center,
Shanghai 200092, China
guoc@sou.edu.cn

³ Management School, Shanghai University of International Business and Economics,
Shanghai 201620, China
daiyonghui@suibe.edu.cn

Abstract. Voice message in social media environment includes a large number of conversation natural languages, which increases the difficulty of emotion tagging and affective computing. In order to solve the above difficulties, this paper analyzes the cognitive differences between the semantic and acoustic features of voice message from the perspective of cognitive neuroscience, and presents a voice feature extraction method based on EEG (Electroencephalogram) experiments, and gets the representation of 25 acoustic feature parameter vectors. Meanwhile, we proposed an affective computing method based on PAD (Pleasure-Arousal-Dominance) dimension emotional space according to the above parameters. Experiments show that the method can effectively solve the affective computing problem of voice message. Overall, there are two main contributions of this paper. Firstly, it comprehensively analyzes the emotional cognitive feature of voice message in social media environment from the perspectives of cognitive neural mechanism, voice acoustic feature and text semantics. Secondly, the segmented affective computing method for voice message based on acoustic feature parameters and PAD emotional state model is proposed.

Keywords: affective feature parameters, cognitive neuroscience, voice acoustic feature, emotion recognition.

1. Introduction

The rise of mobile communication technology and online chat tools has provided the foundation for the development of social media. Wechat, QQ, WhatsApp network communication tools are widely used by people, more and more people exchange information and express opinions through the above tools [32]. The importance of social media in information dissemination and social influence is gradually increasing. From the perspective of communication feature, social media has the feature of fragmented communication content and diversified communication subjects [15]. It is a synthesis of

micro-content, micro-channel and micro-experience. As the most primitive and natural information transmission method for human beings, voice has good convenience and rich emotional feature [30]. Voice message has become a popular way of social communication, and more and more people communicate through the form of voice social communication.

In recent years, the improvement of voice recognition technology has provided support for the promotion of voice social networking. The report released by Baidu company shows that the accuracy of Chinese Mandarin voice recognition is close to 99.8% in a quiet environment, and the technology will be widely used in Baidu voice search product in the future. Because voice not only transmits semantic information, but also contains rich personalized emotional features. The research results and practical cases in recent years have shown that the dissemination of voice message in social media has a huge infectious and synergistic effect, and its emotion has a significant impact on network group cognition, psychology and behavior [3,16]. Great progress has been made in the research of speech emotion recognition in the past few decades. Scholars' research has developed from template matching-based emotion recognition for specific person and simple vocabulary speech to today's statistical model-based emotion recognition for large vocabulary, non-specific person, continuous speech [13, 29]. Many algorithms have been applied to affective computing of speech, such as: K-Means clustering, ant colony, EMD, HMM, SVM algorithms are used in speech recognition [4, 17,].

Previous research provides effective methods and technical tools for solving speech recognition problems in real environments. However, the voice message used in the social media environment such as WeChat, QQ and WhatsApp, which is often completed in the context of the 'small world network' of phrase-based dialogue. The emotion recognition and semantic analysis of voice message are very complicated, and it involves emotional cognitive characteristics. If we only rely on traditional classification methods for affective computing, the effect is not very good. In recent years, EEG technology has made a lot of achievements in cognitive neuroscience research such as the cognitive characteristics of images, voice, text, etc. Therefore, EEG experiments are adapted for our study. In addition, the emotional state of these voice messages is often dynamic, and the previous discrete affective computing methods is not very effective. Therefore, our research has practical significance and good application value.

2. Literature Review

2.1. Affective Feature Parameters of Voice Message and Speech Emotional Model

Some scholars selected ten features such as voice duration, energy, maximum, minimum, median and formant of pitch frequency as emotion recognition features in 2001. Their recognition experimental results of 100 emotion test sentences show that the above features can effectively recognize sadness [40]. Gaussian mixture model was used

to extract the spectral features of speech emotion signals for speech emotion recognition by scholars, their experimental results show that this method has good effect and high recognition efficiency in identifying five emotions: anger, fear, happiness, neutrality and sadness [37]. Some scholars used global control Elman neural network to carry out emotion recognition experiments on speech fused with long-term and short-term features, and achieved a recognition rate of 66.0%. Their experimental research shows that the best recognition segment length of anger, happiness, sadness and surprise is consistent with the prosodic phrase length corresponding to the emotion state [14]. After that, the kernel principal component analysis algorithm, MFCCG-PCA algorithm was proposed. Compared with the general MFCC model, the performance of their model in speech emotion recognition is greatly improved [6].

At present, the existing literature mainly classifies voice emotion based on the traditional six discrete emotions. There are limitations in the analysis of context information and semantics of voice message, and it is difficult to effectively complete emotion recognition in social media environment. Our research is based on the voice emotion recognition, and uses the PAD emotion model of three-dimensional space. The above method can map the emotion types in space, and overcome the discontinuity of emotion classification. It contributes to the research of emotion recognition of voice message in social media environment.

Speech emotion recognition is the ability of a system to recognize human emotions from speech [1]. In order to study speech emotion recognition, phonological affective corpus was proposed, and it was divided into discrete affective corpus and dimensional affective corpus. The former was labeled with language tags, and the latter was labeled with affective spatial coordinates. The representative discrete affective databases with wide influence mainly include: Belfast affective database, Berlin EMO-DB affective phonetic database, FAU AIBO children's German affective phonetic database, CASIA Chinese affective corpus and ACCorpusseries affective database. The above emotional speech database is deduced by the recorder with several kinds of emotions (happiness, anger, neutrality, fear, sadness), which belongs to the performance or guidance emotional database. VAM emotional database selects three emotional dimensions of value, activation and dominion to label [-1, 1]. The database can be used for natural speech understanding and analysis, speech emotion recognition, robust speech recognition research. a fuzzy model of multi-level emotion calculation was constructed according to the emotion dimension to detect various emotions [2]. The recording of Semaine emotion database is carried out in the scene of human-computer interaction, which can be used by researchers free of charge. The audio data is about 7 hours long, sampled at 48Khz, quantized at 24bit, and labeled in five emotional dimensions: value, activation, power, expectation and intensity [26]. Some scholars have studied the PAD emotional model, in which the emotional state is described as a three-dimensional space of Pleasure-Displeasure, Arousal-Nonarousal and Dominance-Submissiveness. So far, PAD emotional model has been widely used in audio-visual speech synthesis, music emotion comparison and speech emotion recognition. In addition, some scholars have studied the speech emotion recognition based on the fusion of HMM and probabilistic neural network. PNN is used to deal with the statistical features in the acoustic feature parameters, HMM is used to deal with the temporal features in the acoustic feature parameters, and then fused through addition and multiplication rules [10].

2.2. Related Research on Brain Cognitive and Neuroscience

Neurocognitive science research shows that the amygdala, prefrontal cortex, hypothalamus and cingulate cortex in human brain structure are related to the processing of emotion, and EEG signals reflect people's emotional state to some extent [22]. In the brain cognitive research of speech emotion, scholars used various emotional speech and music clips as stimulation materials to carry out EEG and ERP experiments and collect data in real time [19, 28]. After denoising, extracting and analyzing the experimental data, they have made a series of important results in the research of speech emotion and brain cognitive function. Some scholars have found that happiness and sadness can cause beta waves changes in the frontal, temporal and parietal regions, and the brain's processing of emotions is related to low-frequency beta waves. In the frontal area, happiness is activated β Wave is more intense than sadness, especially at the frequency of β waves (19.50~25.45 Hz) are the most active [23]. The left frontal lobe related brain area of human brain is easy to be activated by happy and happy speech materials, while the right frontal lobe related brain area is more sensitive to and easier to be activated by fear and sadness speech materials.

The cognitive process of human brain can be reflected by a series of ERP (Event Related Potential) components such as SPN (Stimulus Preceding Negativity), FRN (Feedback Related Negativity) and P300 [31, 35, 41]. Among them, SPN is a continuous and negative slow wave, which responds to attention with expectation, which often appears before the start of task-related stimuli and is closely related to human emotional intelligence [20]. FRN is an important EEG component of brain processing feedback information, and it is mainly distributed in the frontal central region. Its peak appears about 250 milliseconds after the emergence of feedback stimulation, which is closely related to the cognitive evaluation and learning ability of the brain [12]. The latency of P300 component is about 300-600 milliseconds. A large number of experimental studies have found that it is a positive wave of about 300 milliseconds, which is closely related to cognitive processing processes. For example, the allocation of attention resources, the updating process of working memory and inhibition processing [5, 11].

In recent years, the research of cognitive neuroscience has developed rapidly, and its intersection with other disciplines has become a frontier research hotspot. For example: neuro-management, neuromarketing, neuro-entrepreneurship, etc. [9, 33]. The research results of cognitive computing framework and decision-making neural mechanism proposed by scholars on the basis of cognitive neuroscience [34], which provide a reference for our research. One of the innovations and contributions of this paper is to study the cognitive characteristics of the emotional features of speech, which expands the application of cognitive neuroscience.

3. Framework and Method of Voice Affective Computing

3.1. Framework of Voice Affective Computing

In the research of affective computing of voice message, the selection of feature parameters is very important. The cognition of speech information includes not only the cognition of representational information, but also the cognition of semantic information. Among them, the representational information refers to the information that can trigger the rapid initial emotional response of the human brain in a short period of time. Semantic information refers to information that can only produce relatively slow secondary emotional responses after being recognized by the higher cortex of the brain. From the perspective of cognitive neuroscience theory, the physiological structure of human beings receiving speech is the auditory system, which is dominated by the brain, so the cognitive process of speech emotion is the process of the brain's cognition of speech and triggering the corresponding emotional experience. The information conveyed by speech can be divided into two main categories: acoustic information and semantic information [27]. The feature of acoustic information such as pitch, intensity and speed of speech. The semantic information features refer to the characteristics of textual information in speech. Previous studies on brain cognition of speech information have shown differences in cognitive time between acoustic features and word semantic features [21]. With the application of electroencephalography (EEG) in emotion recognition, this paper also uses EEG to perceive emotional features for voice social media. The framework is shown in Fig.1.

In Fig.1, after the voice is preprocessed by phonetic unit interception and voice converted text, affective computing will be performed from both the acoustical features of the voice and the semantic features of the text. Among, speech emotion classifiers mainly include GMM (Gaussian Mixture Model), SVM (Support Vector Machines), HMM (Hidden Markov Model), ANN (Artificial Neural Network) etc. In our study, the combination of SVM and artificial neural network classifier was used for voice emotion classification. There has been a lot of research on the cognition of semantic information in the field of text information communication, but there is still a lack of in-depth research on acoustic feature in the social media environment. This paper studies the response of human's emotional cognition to acoustic features through EEG experiments, so as to provide guidance for the affective computing of voice message and the selection of feature parameters.

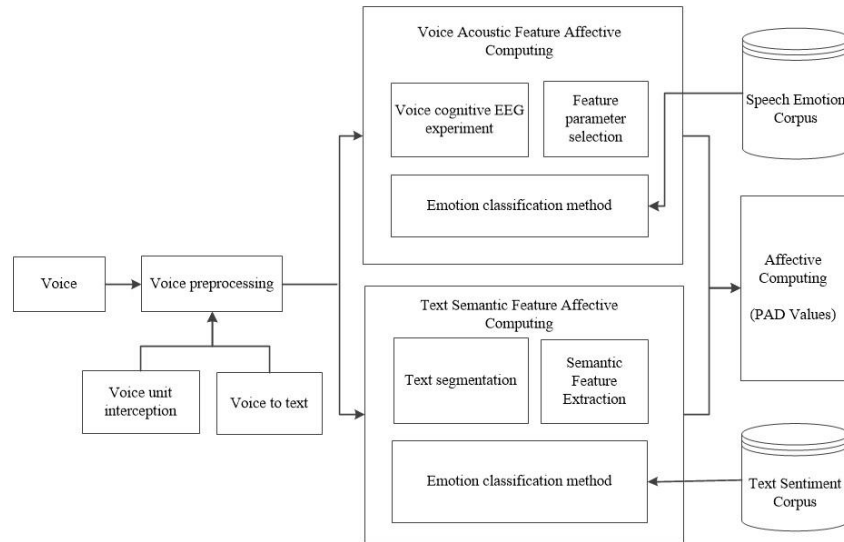


Fig.1. The framework of voice affective computing

3.2. EEG Experiments and Voice Feature Parameters

Considering that the cognition of voice has nothing to do with the identity of subjects, all subjects in EEG experiment come from college students. A total of 15 subjects were recruited in this experiment, aged from 19 to 29 years, with an average of 23.6 years. Among them, 12 (8 males and 4 females) participated in the main experiment and 3 (2 males and 1 female) participated in the control group experiment. All subjects are physically and mentally healthy, right-handed, with normal or corrected-to-normal vision, and no history of mental illness. They had never participated in the EEG experiment before and signed the informed consent before the experiment.

The experimental materials include two parts. Material 1 includes CASIA emotional voice database, which is provided by the human-computer voice interaction research group of the State Key Laboratory of pattern recognition, Institute of automation, Chinese Academy of Sciences. The material is composed of four speakers (two men and two women) who used six emotional types of joy, anger, surprise, sadness, fear and calm, and pronounced 50 sentences once each. It includes 1200 emotional sentence voice, and the sampling rate of each sentence is 16KHz. Material 2 includes 90 units of audio files are collected from social media, film and CD piano music. Each file is divided into audio segments by 8 seconds, and the above segments were labeled with PAD values. The labeled PAD values of these materials are composed of three dimensions: polarity, arousal and dominance, and the range is integer of $[-4, 4]$. Polarity represents happiness or not, arousal means calm or excitement, and dominance refers to a controlled state.

The processing of EEG experiment is shown in Fig.2.

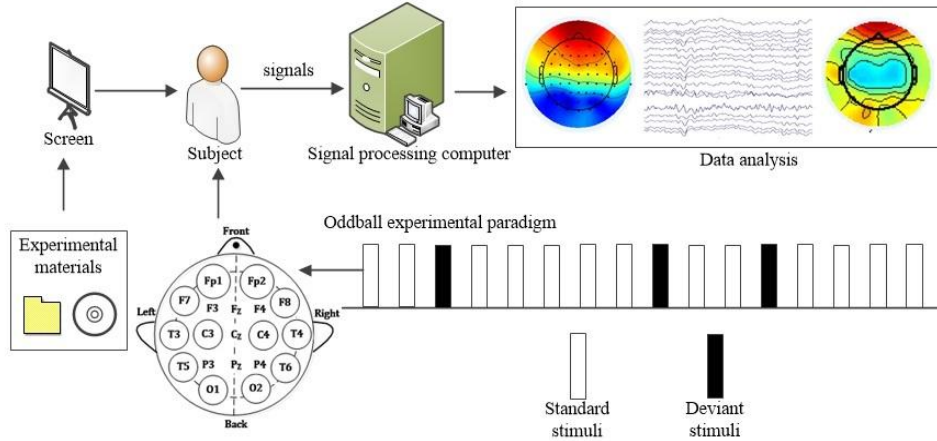


Fig.2. The processing of EEG experiment

In Fig.2, Oddball paradigm is to randomly present two stimuli of the same sensory channel in an experiment, and the probabilities of the two stimuli are very different. The high probability stimuli are called standard stimuli, and the low probability stimuli are called deviant stimuli. In our experimental paradigm, the probability of deviant stimuli is 20%, which meets the criteria of the oddball experimental paradigm.

The experimental process of this study includes four stages as follow: pre-experimental preparation, experimental implementation, post experimental communication, data processing and analysis. Post experimental communication refers to the subjective expression of the experimenter's feelings in the experiment after completing the EEG experiment, such as the emotion at that time and whether it was disturbed. If it was disturbed, the experimental data of the disturbed time segment would be removed in the data processing and analysis stage. According to the feedback form filled by the subjects, combined with their EEG data and voice materials, the corresponding voice characteristics can be gotten by analysis.

In our EEG experiment, the device for collecting data is the product of Shanghai NuoCheng Electric Co., Ltd., and its amplifier model number is NCC Z2N-F-20-C. It adopts wireless WiFi transmission and has strong anti-interference, which has been adopted by many EEG research institutions such as Second Military Medical University. The data sampling rate of the experiment is 128Hz. Because the initial EEG data contains interference components such as EOG (Electro-Oculogram), EMG (Electro-Myogram), it needs to be preprocessed by removing stimulus artifacts, filtering and re-referencing before it can be used for analysis. The sample data of comparison before denoising and after denoising is shown as Fig.3.

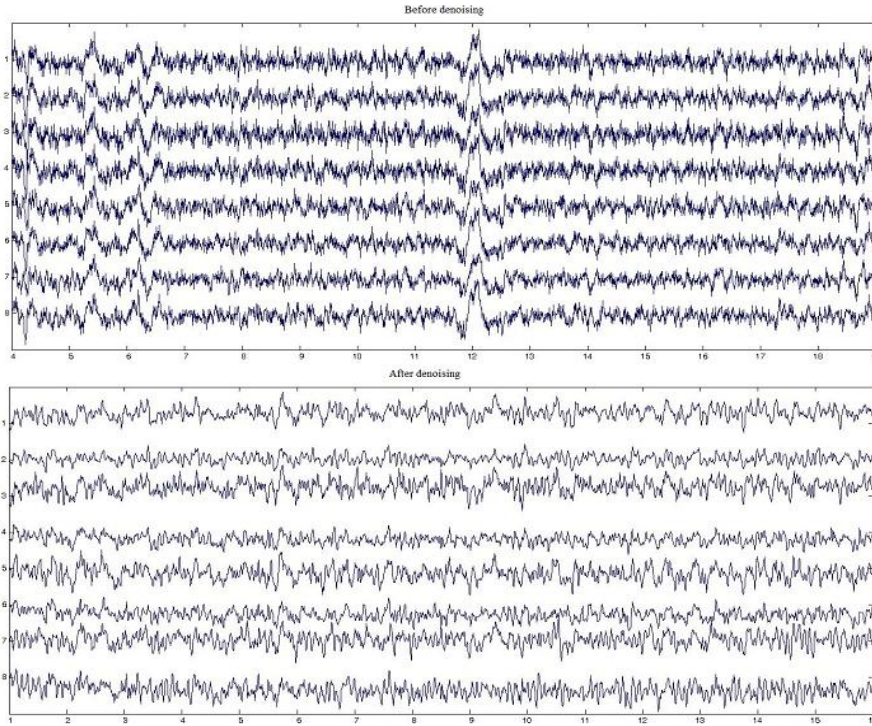


Fig.3. The comparison before and after denoising

After completing the above data preprocessing, we analyzed some electrode point data related to voice cognition by referring to previous research (Li et al. 2018; Yu et al. 2021). The main analysis data came from the signals of eight electrode positions in the frontal, parietal and occipital regions (Fp1, Fp2, T3, T4, C3, C4, O1, O2). Different experimental scenario of brain electrical activity mapping of PSD (Power Spectrum Density) was analyzed, and delta, theta, alpha and beta rhythm is shown as Fig.4.

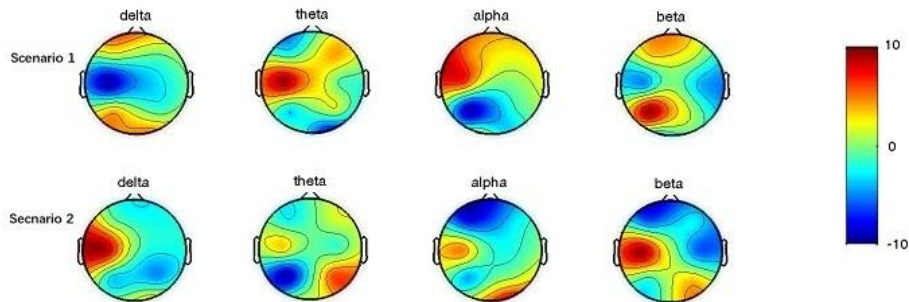


Fig.4. Different experimental scenario of brain electrical activity rhythm

Our experimental results show that beta rhythm waves are more active, especially when they are happy, angry or surprised.

After hearing the fast-speaking or high decibel voice, the power spectral density in the prefrontal, central and occipital regions was significantly more active than in the resting state. Therefore, voice features have an impact on the subjects. After many experiments and analyses, the set of vectors composed of these features can be expressed as follows.

$$I(n) = [P, SE, SZC, MFCC, FF, SF, VS, NVB] \quad (1)$$

In (1), P is the maximum, minimum and average value of the pitch. SE is the maximum, minimum and average values of short-time energy. SZC is the maximum, minimum and average value of the Short Time average Zero-Crossing rate. MFCC is cepstrum coefficient of 12th order Mel frequency. FF is the value of the first formant, SF refers to the value of the second formant. VS is the speed of voice. NVB is the number of breaks in voice. The above value of twenty-five vectors will be used to represent the PAD value.

4. Extraction of Voice Emotion Feature Parameters

In order to get high accuracy in speech feature extraction, it needs to carry out some speech processing, which including voice interception unit, pre-emphasis processing, speech framing and windowing, and end-point detection.

4.1. Voice Signal Pre-emphasis

Voice signal pre-emphasis is mainly to increase the high-frequency part of speech. As the influence of lip pronunciation, the spectrum of high-frequency part of speech will be weakened, resulting in that the main signal left in speech unit information is the spectrum of low-frequency part. In order to avoid this phenomenon, spectrum signal pre-emphasis processing is often adopted. After the speech unit signal is processed by the pre-emphasis filter, it can improve the spectrum of the high-frequency part of the speech signal and filter out the low-frequency interference from 50Hz to 60Hz, which highlights the high-frequency resolution of the speech, so that the random noise can be effectively suppressed in the calculation, which is equivalent to indirectly improving the energy of the voiceless part. The pre-emphasis filter realized by first-order high pass filter is shown as follows.

$$Z(n) = X(n) - a * X(n - 1) \quad (2)$$

In (2), the voice sampling value at the nth time is X(n). a is the pre-weighting coefficient, which is between 0.9 and 1.0, usually is 0.98. The code to realize pre-emphasis in MATLAB software is as follows: `y = filter([1 -1], [1 -0.98], X)`.

4.2. Framing and Windowing

From the whole point of view of speech signal, its feature and parameters characterizing its feature are time-dependent and will change according to time. Previous studies have proved that the gene cycle of voiced voice, the amplitude of voiced voice signal and channel parameters can maintain short-term stability in a short period of time, that is, the spectral feature and some physical characteristic parameters are stable in the above time. Therefore, the speech signal is divided into frames, that is, the voice signal is divided into time periods in frames, and the frame length is generally 10ms ~ 30ms. The non-overlapping part between frames is called frame shift, which is generally set to 50%. After the above processing, the analysis and processing of speech stabilization process can be performed. The framing operation can be completed through the Enframe() function in MATLAB. In this function, the Len() parameter is the frame length (10~30ms) and the Inc parameter is the frame shift (1/3~1/2). In order to keep the emotional signal of speech unit stable in a short time, window function is usually used to process the signal. The commonly used window functions are rectangular window, Hamming window and Hanning window. The mathematical formulas of the three window functions are shown as follows.

$$W(n) = \begin{cases} 1, & 0 \leq n \leq N - 1 \\ 0, & \text{Other} \end{cases} \quad (3)$$

In (3), it represents a rectangular window, where n is the window length, which is usually represented by frame length.

$$W(n) = \begin{cases} 0.54 - 0.46 \cos[2\pi n / (N - 1)], & 0 \leq n \leq N - 1 \\ 0, & \text{Other} \end{cases} \quad (4)$$

$$W(n) = \begin{cases} 0.5[1 - \cos(2\pi n / (N - 1))], & 0 \leq n \leq N - 1 \\ 0, & \text{Other} \end{cases} \quad (5)$$

Rectangular window is usually used in the time domain of speech processing. Among, Hamming window or Hanning window is often selected in the frequency domain of speech processing. The code to realize Hamming window in MATLAB software is as follows: $X = X' * \text{Hamming}(n)$, where n is the number of data points of each window.

4.3. End-Point Detection

End-point detection is to find the starting point and ending point of a speech. As an important method of pre-processing speech data, Voice Activity Detection (VAD) functions was used to detect presence or absence of human voice in a signal [1]. After the voice is detected by VAD, the invalid voice is removed, and the remaining voice is used for subsequent processing and analysis. In our work, short-term average energy and short-term average zero crossing rate was used to VAD.

Short time average energy. After speech signal is divided into frames, the characteristic parameter used in time domain processing is short-term average energy, which is used to distinguish voiced segment from unvoiced segment. The short-term average energy is described as follows. If $x(n)$ is the time domain signal of speech waveform and $y_i(n)$ is the i -th frame signal of windowed function $w(n)$, then $y_i(n)$ is calculated as follows.

$$y_i(n) = w(n) * x((i-1) * inc + n), \quad 1 \leq n \leq L, 1 \leq i \leq fn \quad (6)$$

In (6), Where $w(n)$ is the window function, $y_i(n)$ is the value of the frame, FN and L refers to the frame length, inc is the frame shift length.

Therefore, the short-time average energy formula of the i -th frame speech signal $y_i(n)$ is calculated as follows.

$$E(i) = \sum_{n=0}^{L-1} y_i^2(n), \quad 1 \leq i \leq fn \quad (7)$$

Short time average zero crossing rate. The short-term average zero crossing rate refers to the number of times each frame signal passes through the zero value. It is used to roughly estimate the spectral feature of speech signals. For discrete-time speech signals, the zero-crossing phenomenon means that the algebraic symbols of two adjacent sampled values are different; For continuous speech signal, it should be judged according to how its time-domain waveform crosses the horizontal axis. The formula of short-time average zero crossing rate is calculated as follows.

$$Z(i) = \frac{1}{2} \sum_{n=0}^{L-1} |\text{sgn}[y_i(n)] - \text{sgn}[y_i(n-1)]|, \quad 1 \leq i \leq fn \quad (8)$$

In (8), $\text{sgn}[]$ is a symbolic function, which is calculated as follows.

$$\text{sgn}[x(n)] = \begin{cases} 1, & x(n) \geq 0 \\ -1, & x(n) < 0 \end{cases} \quad (9)$$

Mel-Frequency Cepstral Coefficients. MFCCs is a set of audio characteristic parameters. This coefficient uses the auditory principle and cepstrum solution to recognize speech. It is a parameter established by imitating the auditory feature of human ears and has high noise resistance. The conversion relationship between MFCC coefficient and linear frequency is calculated as follows.

$$f_{Mel} = 2595 * \log \left[1 + \frac{f}{700} \right] \quad (10)$$

In (10), f is the frequency. MFCC can be used as the frequency sensitivity of human hearing. Human ears show logarithmic changes in frequency perception. The MFCC parameters of order 12 can be obtained by using the function `melbankm` (m, N, FS) in the voicebox of MATLAB toolbox, where m parameter is the number of filters, n parameter is the speech frame length, and FS parameter is the sampling frequency. The 12th order MFCC parameter feature of the sample voice file is shown in Fig.5.

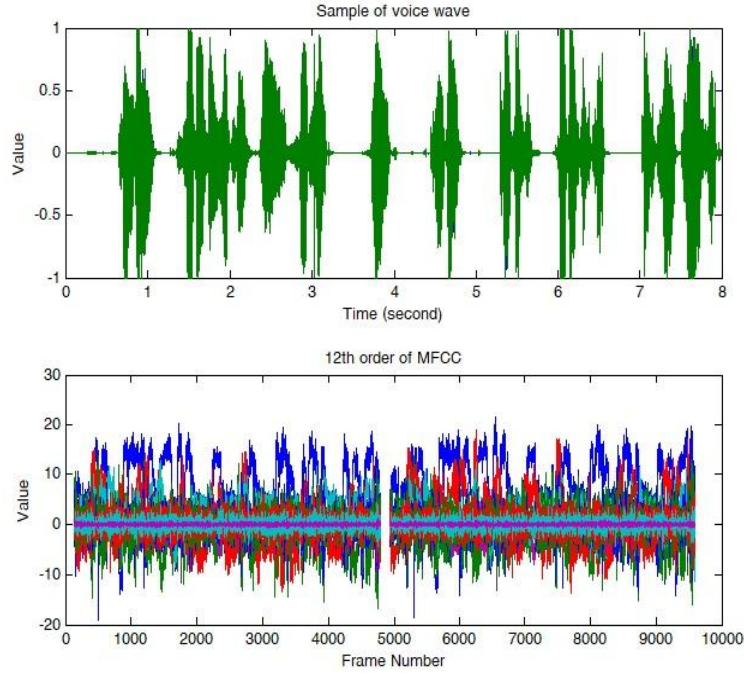


Fig.5. 12th order MFCC feature parameters of sample voice

4.4. PAD Emotional State Model

In the PAD (Pleasure-Arousal-Dominance) emotional state model, the emotional state is described through three-dimensional space. Among them, the positive and negative changes of emotional state are expressed by the score from Pleasure to Displeasure. The emotional physiological activation level and alertness changes are expressed by the scores of Arousal to Nonarousal. The control and influence of emotions on other individuals and the external environment are expressed by Dominance to Submissiveness scores [8, 25]. The calculation of PAD value in this paper is based on the PAD questionnaire scale provided by the Chinese Academy of Sciences. The scale includes three dimensions: pleasure, activation and dominance. Each dimension has four items, which is a 9-point semantic difference scale [39]. Questionnaire scale of PAD emotional state model is shown in Table 1.

Table 1. Questionnaire scale of PAD emotional state model

Question	Emotional state	Value	Emotional state
Q1	Angry	-4, -3, -2, -1, 0, 1, 2, 3, 4	Activated
Q2	Wide awake	-4, -3, -2, -1, 0, 1, 2, 3, 4	Sleepy
Q3	Controlled	-4, -3, -2, -1, 0, 1, 2, 3, 4	Controlling
Q4	Friendly	-4, -3, -2, -1, 0, 1, 2, 3, 4	Scornful
Q5	Calm	-4, -3, -2, -1, 0, 1, 2, 3, 4	Excited
Q6	Dominant	-4, -3, -2, -1, 0, 1, 2, 3, 4	Submissive
Q7	Cruel	-4, -3, -2, -1, 0, 1, 2, 3, 4	Joyful
Q8	Interested	-4, -3, -2, -1, 0, 1, 2, 3, 4	Relaxed
Q9	Guided	-4, -3, -2, -1, 0, 1, 2, 3, 4	Autonomous
Q10	Excited	-4, -3, -2, -1, 0, 1, 2, 3, 4	Enraged
Q11	Relaxed	-4, -3, -2, -1, 0, 1, 2, 3, 4	Hopeful
Q12	Influential	-4, -3, -2, -1, 0, 1, 2, 3, 4	Influenced

The scale and the normalized PAD values are calculated as follow.

$$P = \frac{Q1 - Q4 + Q7 - Q10}{16} \quad (12)$$

$$A = \frac{-Q2 + Q5 - Q8 + Q11}{16} \quad (13)$$

$$D = \frac{Q3 - Q6 + Q9 - Q12}{16} \quad (14)$$

5. Experimental Verification

5.1. Experimental Case

In our research, we take the view of “Shanghai COVID-19” as an experiment in WeChat’s speech transmission. 80 voice chat records about the event were collected from WeChat group. An example of text converted from a sample voice is as follows. “The materials received today include vegetables, meat and eggs. Great! Thank you very much, thank the volunteers!”.

5.2. Measurement of Acoustic Feature Parameters

A total of 25 vectors of acoustic feature parameters are extracted, namely: short-term energy (max, min, mean), pitch (max, min, mean), short-term average zero crossing rate

(max, min, mean), first formant, second formant, speech speed, number of interlanguages pauses and Cepstrum Coefficient of 12th order Mel frequency standard. Fig.6 shows the emotion recognition rate of the test results.

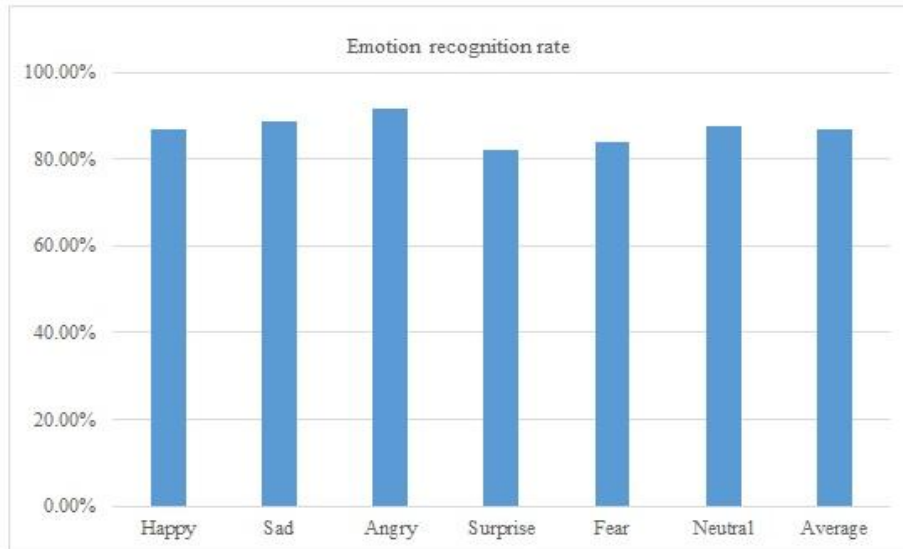


Fig.6. Emotion recognition rate

The results show that the recognition rates of six categories of emotions: happy, sad, anger, surprise, fear and neutral are 86.69%, 88.54%, 91.71%, 81.93%, 83.82% and 87.73%, and the average recognition rate is 86.74%.

In order to extract the voice social parameters of personalized features, we take 80 sentences from the historical voice of WeChat group. The above sentences come from 8 person, with an average of 10 sentences each person. From previous studies, it can be seen that social media voice includes a large number of short sentences, and the changes in emotional states exhibit dynamic characteristics [10]. Therefore, based on previous studies and the analysis of voice social habits, 8 seconds is used as the interval period for processing voice message. Each voice is sampled 8 seconds and trained with BP neural network. After reaching the accuracy requirements ($\epsilon < 0.001$), the trained BP neural network is retained for the next experiment. The sample voice feature parameters and PAD values are shown in Table 2.

It can be seen from Table 2 that the PAD values are composed of three values (0.42, 0.69, 0.21). Because P value is between 0.4 and 0.53, A value is between 0.67 and 0.73, and D value is between 0.17 and 0.3, its emotion type is happiness according to the distribution range of PAD value of the six emotion types [7].

Table 2. The sample voice feature parameters and PAD values

Acoustic characteristic parameters	Pitch (Max/Min/Mean)	Short time energy (Max/Min/Mean)	Short time average zero crossing rate (Max/Min/Mean)	12th order MFCC	First formant	Second formant	Speed of voice	number of breaks in voice	PAD values
Duration (8S)	190.316	86.971	431	21.7303	799.14	1937.28	0.125/s	18	P=0.42 A=0.69 D=0.21
	76.857	32.528	32	13.8392					
	122.534	76.542	224.544	18.9420					
				12.6046					
				6.0004					
				1.0265					
				10.6584					
				12.7003					
				13.0288					
				7.7237					
				5.7858					
				1.2557					

6. Discussion and Conclusion

Although speech emotion recognition has made great progress in the past few decades, the emotion calculation of continuous and short-time conversational natural language speech signals in social media still faces many challenges. Especially the calculation of complex mixed emotion and its dynamic change process is still a difficult problem to be explored. In order to solve the above problems, this paper refers to the existing research results and proposes a framework for feature extraction and affective computing of voice message in social media environment. Specifically, it includes voice acoustic feature effective computing and text semantic feature effective computing. This framework is based on EEG experiment and PAD emotion model, which is the main innovation of this paper. In this framework, the emotion of the sample voice is calculated, and the recognition accuracy is 86.74%. The experimental results show that it provides an effective method for computing voice emotion in social media environment.

In general, the contributions of this paper are mainly in two aspects: on the one hand, the emotional cognitive feature parameters of voice message are given. The above parameters have good universality because their extraction is completed on the basis of EEG experiments. On the other hand, a new solution is proposed to transform voice affective computing into PAD emotion model parameter estimation based on acoustic parameters, which provides a reference for the study of feature extraction and affective computing of voice message in social media environment.

In the future, our study will be improved with more sample analysis, cognitive neuroscience experiment and research tools. For example: it combines fMRI (Functional Magnetic Resonance Imaging), FNIRS (Functional near-infrared spectroscopy) instruments for study.

Acknowledgment. This work is supported by the project of Social Science Foundation of Fujian Province of China (No. FJ2022BF033), and Shanghai Open University discipline innovation in 2019 year (No. XK1904).

References

1. Alghifari, M.F., Gunawan, T.S., Nordin, M.A.W., Qadri, A.A.A.: Kartiwi, M., Janin, Z. H.: On the use of voice activity detection in speech emotion recognition. *Bulletin of Electrical Engineering and Informatics*, Vol. 8, No. (4), 1324-1332. (2019)
2. Bakhtiyari, K., Husain, H.: Fuzzy model of dominance emotions in affective computing. *Neural Computing & Applications*, Vol. 25, No. 6, 1467-1477. (2014)
3. Bänziger T., Patel, S., Scherer, K.: The Role of Perceived Voice and Speech Feature in Vocal Emotion Communication. *Journal of Nonverbal Behavior*, Vol. 38, No. 1, 31-52. (2014)
4. Bhagavathsingh, B., Srinivasan, K., Natrajan, M.: Real time speech based integrated development environment for C program. *Circuits and Systems*, 7(3), 69-82. (2016).
5. Challawar, R., Menon. A., Kar, M., Mahapatra, S.C.: Effect of acute bout of moderate exercise on P300 component of event-related potential in young women during different phases of menstrual cycle: A pilot study. *Indian Journal of Physiology and Pharmacology*, Vol. 64, No. 4, 272-278. (2021) .
6. Chen, W.L., Sun, X.: Mandarin Speech Emotion Recognition Based on MFCCG-PCA. *Acta Scientiarum Naturalium Universitatis Pekinensis*, Vol. 51, No. 2, 269-274. (2015)
7. Chen, Y.L., Cheng, Y.F., Chen, X.Q., Wang, H.X., Li, Chao.: Speech emotion estimation in PAD 3D emotion space. *Journal of Harbin Institute of Technology*, Vol. 50, No. 11, 160-166. (2018).
8. Choe, Kyung-II.: An Emotion-Space Model of Multimodal Emotion Recognition. *Advanced Science Letters*, Vol. 24, No. 1, 699-702. (2018)
9. Dai, W.H.: Neuromangement: disciplinary development and research paradigm. *Journal of Beijing Technology and Business University (Social Sciences)*, Vol. 32, No. 4, 1-10. (2017)
10. Dai, W.H., Han, D.M., Dai, Y.H., Xu, D.R.: Emotion recognition and affective computing on vocal social media. *Information & Management*, Vol. 52, No. 7, 777-788. (2015)
11. Dehaene, S.: The Error-Related Negativity, Self-Monitoring, and Consciousness. *Perspectives on Psychological Science*, Vol. 13, No. 2, 161-165. (2018)
12. Gerstner, W., Sprekeler, H., Deco, G.: Theory and Simulation in Neuroscience. *Science*, Vol. 338, No. 6103, 60-65. (2012)
13. Gong, S.P., Dai, Y.H., Ji, J., Wang, J.Z., Sun, H.: Emotion Analysis of Telephone Complaints from Customer Based on Affective Computing. *Computational Intelligence and Neuroscience*, 2015, 1-9. (2015).
14. Han, W.J., Li, H.F., Han, J.Q.: Speech emotion recognition with combined short- and long-term features. *Journal of Tsinghua University (Science and Technology)*, Vol. S1, 708-714. (2008)
15. Hu, J.: Building a risk prevention system to protect the mainstream ideology in the era of micro communication. *Journal of Dalian University of Technology (Social Sciences)*, Vol. 42, No. 3, 1-6. (2021)
16. Huang, S., Zhou, X., Xue, K., Wan, X.Q., Yang, Z.Y., Xu, D., Ivanović, M., Yu, X.: Neural Cognition and Affective Computing on Cyber Language. *Computational Intelligence & Neuroscience*, Vol. 2015, 1-10.

17. Jalili, F., Barani, M.J.: Speech Recognition Using Combined Fuzzy and Ant Colony algorithm. *International Journal of Electrical & Computer Engineering*, Vol. 6, No. 5, 2205-2210. (2016)
18. Jiang, N., Liu, T.: An Improved Speech Segmentation and Clustering Algorithm Based on SOM and K-Means. *Mathematical Problems in Engineering*, Vol. 1, 1-19. (2016)
19. Kstle, J.L., Anvari, B., Krol, J., Wurdemann, H.A.: Correlation between Situational Awareness and EEG signals. *Neurocomputing*, Vol. 432, No. 1, 70-79. (2021)
20. Kumar, J., kumar, J.: Affective Modelling of Users in HCI Using EEG. *Procedia Computer Science*, Vol. 84, No. 5, 107-114. (2016)
21. Kurbalija, V., Ivanovic, M., Radovanovic, M., Geler, Z., Dai, W.H., Zhao, W.D.: Emotion Perception and Recognition: An Exploration of Cultural Differences and Similarities. *Cognitive Systems Research*, Vol. 52, 103-116. (2018)
22. Lai, Y., Tian, Y., Yao, D.: MMN evidence for asymmetry in detection of IOI shortening and lengthening at behavioral indifference tempo. *Brain Research*, Vol. 1367, No. 7, 170-180. (2011).
23. Lee, K.J., Park, C.A., Lee, Y.B., Kim, H.K., Kang, C.K.: EEG signals during mouth breathing in a working memory task. *International Journal of Neuroscience*, Vol. 130, No. 5, 1-10. (2019)
24. Li, H.W., Li, H.F., Ma, L., Bo, H.J., Xu, R.F.: Brain's cognitive law of changes in musical attributes while listening to music-An EEG study. *Journal of Fudan University (Natural Science)*, Vol. 57, No. 3, 385-392. (2018)
25. Li, J., Huang, W., Guo, S.L., Sun, Y.: Research on the Sentiment Intensity Measurement Model of Internet Word-of-Mouth Public Opinion Based on the PAD Model. *Journal of the China Society for Scientific and Technical Information*, Vol. 38, No. 3, 277-285. (2019)
26. Lin, J.C., Wu, C.H., Wei, W.L.: Error Weighted Semi-Coupled Hidden Markov Model for Audio-Visual Emotion Recognition. *IEEE Transactions on Multimedia*, Vol. 14, No. 1, 142-156. (2012)
27. Liu, Z.T., Xu, J.P., Wu, M., Cao, W.H., Chen, L.F., Ding, X.W., Hao, M., Xie, Q.: Review of emotional feature extraction and dimension reduction method for speech emotion recognition, *Chinese Journal of Computers*. Vol. 41, No. 12, 2833-2851. (2018)
28. Ma, Q.G., Feng, Y.D., Xu, Q., Bian, J., Tang, H.X.: Brain potentials associated with the outcome processing in framing effects. *Neuroscience letters*, Vol. 528, No. 2, 110-113. (2012)
29. Mcaleavy, T., Rhisart, M.: Harnessing the power of metaphor: uncovering a hidden language of interoperability within the natural speech of emergency managers. *International Journal of Emergency Management*, Vol. 15, No. 1, 1-25. (2019)
30. Motamed, S., Setayeshi, S., Rabiee, A.: Speech emotion recognition based on brain and mind emotional learning model. *Journal of Integrative Neuroscience*, Vol. 17, No.12, 1-15. (2018)
31. René, R., Mohr, P.N.C., Kenning, P.H., Davis, F.D., Heekeren, H.R.: Trusting Humans and Avatars: A Brain Imaging Study Based on Evolution Theory. *Journal of Management Information Systems*, Vol. 30, No. 4, 83-113. (2014)
32. Shankar, S., Tewari, V.: Understanding the Emotional Intelligence Discourse on social media: Insights from the Analysis of Twitter. *Journal of Intelligence*, Vol. 9, No. 4, 1-17. (2021)
33. Sharma, G.D., Paul, J., Srivastava, M., Yadav, A., Mendy, J., Sarker, T., Bansal, S.: Neuroentrepreneurship: an integrative review and research agenda. *Entrepreneurship and Regional Development*, Vol. 33, 863-893. (2021)
34. Song, X.Y., Zeng, Y., Tong, L., Shu, J., Li, H.M., Yan, B.: Neural Mechanism for Dynamic Distractor Processing during Video Target Detection: Insights from Time-varying Networks in the Cerebral Cortex. *Brain Research*, Vol. 1765, 1-9. (2021)

35. Wang, H.L., Feng, T.Y., Suo, T., Liang, J., Meng, X.X., Li, H: The process of counterfactual thinking after decision-making: Evidence from an ERP study. *Chinese Science Bulletin*, Vol. 55, No. 12, 1113-1121. (2010)
36. Yu, G.M., Wang, W.X., Feng, F., Xiu, L.C. Evaluation of the communication effect of synthetic speech news: The EEG evidence of the effect of speech speed. *Chinese Journal of Journalism & Communication*, Vol. 43, No. 2, 6-26. (2021)
37. Yun, S., Yoo, C.D.: Loss-Scaled Large-Margin Gaussian Mixture Models for Speech Emotion Classification. *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 20, No. 2, 585-598. (2012).
38. Zhang, G.: Quality evaluation of English pronunciation based on artificial emotion recognition and gaussian mixture model. *Journal of Intelligent and Fuzzy Systems*, Vol. 40, No. 2, 1-11. (2020)
39. Zhang, X.Y., Zhang, T., Sun, Y., Zhang, W., Chang, J.: Emotional Speech Database Optimization and Quantitative Annotation Based on PAD emotional model. *Journal of Taiyuan University of Technology*, Vol. 48, No. 3, 469-474. (2017)
40. Zhao, L., Qian, X.M, Zhou, C.R., Wu, Z.Y.: A Study on Emotional Recognition in Speech Signal. *Journal of Software*, Vol. 7, 1050-1054. (2001)
41. Zheng, J.: Cheng, J., Wang, C., Lin, X., Fu, G., Sai, L.: The effect of mental countermeasures on a novel brain-based feedback concealed information test. *Human brain mapping*, Vol. 43, No. 9, 2771-2781. (2022)

Peng Jiang is an associate professor at Jingan Branch Campus, Shanghai Open University, China. His current research interests include Educational technology and Management Information System. Contact him at jzhpmail@163.com.

Cui Guo is a lecturer at Shanghai Lifelong Education School Credit Bank Management Center, China. Her current research interests include Distance Learning and Computer Application System. Contact her at guoc@sou.edu.cn.

Yonghui Dai is the corresponding author of this paper. He is currently an associate professor at the Management School, Shanghai University of International Business and Economics, China. He received his Ph.D. in Management Science and Engineering from Shanghai University of Finance and Economics, China in 2016. His current research interests include Affective Computing, Intelligence Service and Big Data Analytics. His works have appeared in international journals more than forty papers. Contact him at daiyonghui@suibe.edu.cn.

Received: May 09, 2023; Accepted: September 15, 2023.

Machine Learning and Text Mining based Real-Time Semi-Autonomous Staff Assignment System

Halil Arslan¹, Yunus Emre Işık², Yasin Görmez², and Mustafa Temiz²

¹ Department of Computer Engineering, Sivas Cumhuriyet University
58140 Sivas, Türkiye
harslan@cumhuriyet.edu.tr

² Department of Management Information Systems, Sivas Cumhuriyet University
58140 Sivas, Türkiye
{yeisik,yasingormez,temizmustafa}@cumhuriyet.edu.tr

Abstract. The growing demand for information systems has significantly increased the workload of consulting and software development firms, requiring them to manage multiple projects simultaneously. Usually, these firms rely on a shared pool of staff to carry out multiple projects that require different skills and expertise. However, since the number of employees is limited, the assignment of staff to projects should be carefully decided to increase the efficiency in job-sharing. Therefore, assigning tasks to the most appropriate personnel is one of the challenges of multi-project management. Assigning a staff to the project by team leaders or researchers is a very demanding process. For this reason, researchers are working on automatic assignment, but most of these studies are done using historical data. It is of great importance for companies that personnel assignment systems work with real-time data. However, a model designed with historical data has the risk of getting unsuccessful results in real-time data. In this study, unlike the literature, a machine learning-based decision support system that works with real-time data is proposed. The proposed system analyses the description of newly requested tasks using text-mining and machine-learning approaches and then, predicts the optimal available staff that meets the needs of the project task. Moreover, personnel qualifications are iteratively updated after each completed task, ensuring up-to-date information on staff capabilities. In addition, because our system was developed as a microservice architecture, it can be easily integrated into companies' existing enterprise resource planning (ERP) or portal systems. In a real-world implementation at Detaysoft, the system demonstrated high assignment accuracy, achieving up to 80% accuracy in matching tasks with appropriate personnel.

Keywords: multi-project management, task assignment, text mining, staff assignment system

1. Introduction

The continuous development of technology and information systems is rapidly changing the business mentality in the global world. Companies have started to use information systems, customized applications, and software in all departments such as production, logistics, marketing and human resources in order to reduce costs, gain a competitive advantage and make the organization more efficient. The need to transform and update existing business processes in companies using information-based systems has emerged with

the introduction of public obligations. However, efforts to meet Information Technology (IT) requirements in large organizations with internal resources may not be sustainable, as complex projects require different levels of expertise. Instead, outsourcing projects to consulting firms that have expert staff offers benefits in terms of focusing on the main area of work, increasing quality, and reducing costs [29].

On the other hand, the workload of consulting firms has increased as the number of clients with diverse businesses who are awaiting the services of their consultants for the new project increased. In addition, needs analysis, action plan, resource allocation, and testing procedures should be scheduled for the requested project/task support [13]. Timely completion of project tasks requiring different skills is also possible only with multi-project management (MPM).

MPM is an approach that involves planning, executing, monitoring, and completing multiple projects simultaneously using the same set of human resources. The objective of this management approach is to achieve optimal organizational performance by effectively balancing and coordinating projects that require specialized expertise while dealing with limited resources[11]. In cases where the most important resource is educated and qualified personnel, the optimal output can only be achieved by assigning the right personnel to the right project. This challenge is called a task assignment problem in the MPM environment. Task assignment is about finding the most suitable personnel who have the required skills to perform the tasks in a new or existing project. When determining personnel, it is expected that the employee's qualifications match the requirements of the task. Otherwise, the non-appropriate assignment of personnel might lead to delays in projects. Since projects in IT consulting are generally software or development tasks, each of them must be carried out by a qualified person according to the needs of the task. Even if there are many employees in a company, not every employee may have the same skills and experience. Therefore, there is a need for solutions that efficiently facilitate task assignment is crucial for enhancing productivity and maximizing the utilization of qualified human resources.

The existing studies in the literature treat the assignment of tasks and employees in MPM as scheduling and optimization of resources depending on various objective functions [15], [27],[24],[16]. Besides there are some studies that tried to solve real problems of companies. Lie et al. proposed the Critical Chain Project Management approach to efficiently manage projects in research centers. In this approach, resources are divided into small parts, and procurement of required equipment is included in the project management process, which increases efficiency [20]. Chen et al. proposed an integrated model for scheduling multiple projects and assigning staff with multiple skills for IT products. They defined four objectives to be considered simultaneously: Skill efficiency, product development, time and cost. The non-dominated Sorting Genetic Algorithm II is used to solve the optimization problems [8].

One of the most important challenges of MPM is the problem of efficient task assignment [17]. Task assignment is about finding the most suitable personnel who have the required skills to perform the tasks in a new or existing project. When determining personnel, it is expected that the employee's qualifications match the requirements of the task. In a study, Cai and Li proposed a genetic algorithm-based model for assigning employees with multiple skills to the right tasks [5]. Cheng and Chu proposed a model that considers employee qualifications and optimizes the task assignment problem using fuzzy set the-

ory and genetic algorithm [9]. Almost all of these studies aim to demonstrate the success of optimization models in theoretical terms by testing them on fixed datasets. Since these approaches rely on constraints, they must be repeated whenever a new task is requested. Almost all projects run by companies have a deadline. Projects that cannot be completed by the deadline cause companies to incur huge losses. The fastest way to assign appropriate employee, which is one of the most important factors for projects, also significantly affects the timely completion of the project. In this context, task assignments should be completed immediately when a new task is requested. The re-optimization of the models can take a lot of time depending on the size of the data used. Considering that decision makers need to make the assignment process quickly, the models in the literature are not suitable for a real-time system. Contrary to this situation, there is no need for re-training in the system that was proposed in this study.

A machine learning solution incorporates a text-mining approach, on the other hand, can be used to recommend efficiently a staff once it has been properly trained. After all, given that project tasks are identified and described through textual information, text mining emerges as one of the most effective ways for analyzing such textual data and deriving valuable insights from it. Text mining, also known as text analysis or text data mining, is a field that combines multiple disciplines to extract valuable information and knowledge from unstructured text data [30]. Utilizing pre-processing techniques and algorithms, enables automatic processing and interpretation of given text data, leading to several advantages over manual content analysis, such as less required time and human work needed. [14]. Mo et.al. proposed a model to improve the productivity of staff assignments using text mining-based machine learning techniques. More than 82,000 collected maintenance records from different universities were vectorized using Bag-of-Words (BOW). Logistic Regression, Support Vector Machines, and Naive Bayes methods are used to train the datasets. The models are then used to determine the personnel for specific tasks. As a result, 77% of the tasks are correctly assigned to the correct personnel [23]. A similar machine learning-based study was conducted to route service requests to the help desk system to the correct staff. In the study, using algorithms such as SVM, Decision Tree, and Naive Bayes, service request tickets are classified and routed to the right staff with 81% Accuracy [3].

Appropriate personnel assignment studies are directly related to companies working on more than one project at the same time. It is possible that decision support systems designed using datasets created from historical data may give worse results in real-time data. In this context, the fact that a study on personnel assignment also works with real-time data of companies is one of the important factors that increase the value of the study. Especially for companies operating in the field of consultancy, assigning personnel using real-time data has crucial importance. Companies that manage multiple projects are generally medium or large-sized companies, and these companies mostly run their projects through ERP systems. In this study, real-time data obtained from the ERP system of a software consultancy firm operating in Turkey were used. The performance obtained using real-time data with the proposed model in this study has similar results with the analyzes made using historical data in the literature. In fact, better results have been obtained from many models that analyze using historical data in the literature. Considering this structure, our study differs positively from the studies in the literature.

2. Materials and Methods

2.1. Grounded Theory

In qualitative research methods, the establishment of theories about data can lead to the collection of specific data that validate that theory. If there are not enough theories in the literature on a topic, deductive reasoning may not be the best way to find out true theories about data. In order to draw meaningful conclusions about the collected data on any topic, theories should be made about the existing data. In statistical analysis, this approach is called grounded theory. Grounded Theory (GT) is a qualitative research method based on the systematic construction of hypotheses and theories through the collection and analysis of qualitative data [7]. It provides summarized ideas and concepts from the collected data. GT follows 4 iterative rules: Finding representative concepts by reviewing the data, recoding concepts using keywords, hierarchically grouping the codes into concepts, and categorizing them by similarity. At the end of the iterative analysis, the categories and the connections between them are used as theory.

2.2. TF/TFIDF

The most commonly used text vectorization methods in the literature are Term Frequency (TF) and Term Frequency-Inverse Document Frequency Ratio (TF-IDF). TF simply indicates the ratio of the occurrence of each word to the total number of words and is calculated by dividing the frequency of occurrence of a word by the number of words in the document [22]. Thus, if the ratio of a word is higher, it is considered as an important and prominent word with respect to the topic of the document. On the other hand, TF-IDF is calculated by multiplying the inverse document frequency, which indicates the importance coefficient of a word in a given document, by the term frequency. Thus, TF-IDF considers not only the word frequency but also the importance coefficient of a word in a document to indicate the coefficient. Since these approaches are most commonly used, we tested them in our dataset and compared them with word embedding methods [31].

2.3. Word2Vec

Word2Vec, which is a word embedding approach, is a technique that maps words to a vector of numbers. It uses a two-layer neural network model to understand the semantic relationship behind words in a given context, and hypothesizes that words with close meaning also have close vectorial distance [21]. This vectorization method has 2 different learning algorithms, CBOW (Continuous Bag of Words) and Skip-Gram, as shown in figure 1.

CBOW receives a couple of words $W_n = W_t - 2, W_t - 1, W_t + 1, W_t + 2$ as input, where n denotes the window size of words and W_t denotes the target words. The main principle of CBOW is to predict a particular word by analyzing the neighboring words. On the other hand, Skip-Gram attempts to predict the surrounding words from the target words, as a reversal of CBOW. Skip-Gram takes advantage of vectorization when a new word appears in context. The projection layer in both models is an N -dimensional vector projected by an encoded input vector. This layer stores a single set of common weights and therefore projects all words to the same position.

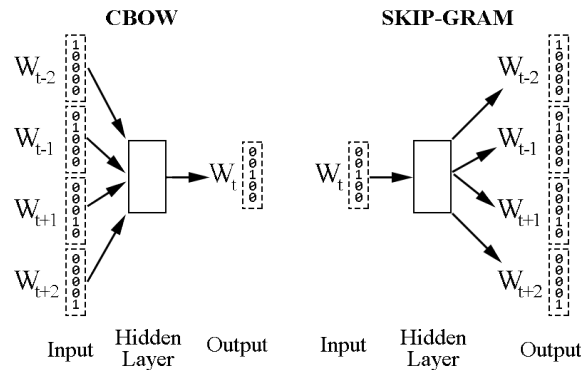


Fig. 1. Learning Models of Word2Vec Representation. The 't' represents the current word. In CBOW, a window of context is employed to predict the target word by considering both preceding and succeeding words. In contrast, Skip-Gram aims to predict the previous and next words using the middle word as the context

2.4. Doc2Vec

The Word2Vec method attempts to discover similarity between words by using co-occurrence frequencies and vector distances, but the Doc2Vec approach represents the entire document or paragraph in a vector space rather than just words, regardless of text length [19]. Therefore, it is considered as an extension of the Word2Vec approach. In this method, additional identifier information is added to the vector so that paragraph-to-paragraph relationships can be learned and models can understand the similarities between paragraphs behind words.

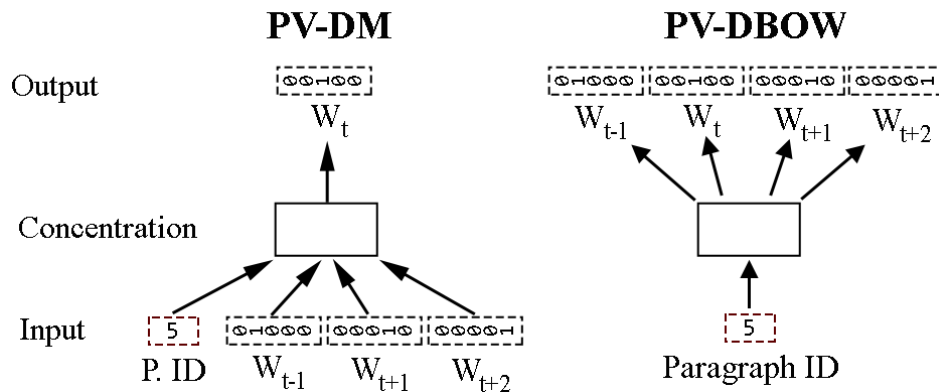


Fig. 2. Learning Approaches of Doc2Vec Representation. Similar to Word2Vec learning but includes an extra information paragraph id

The Doc2Vec representation method includes PV-DM (Paragraph Vector-Distributed Memory) and PV-DBOW (Paragraph Vector-Distributed Bag of Words) like the CBOW and Skip-Gram learning algorithms. PV-DM enables paragraph-specific word learning by using the unique identity value of the paragraph along with the word vectors. On the other hand, PV-DBOW does not take a word vector but the unique identification number of the paragraph and tries to predict the target words (see Figure 2).

2.5. FastText

FastText is a text mining library developed by Facebook group for text classification and word representation. The algorithm behind FastText is slightly different from other word representation methods such as Word2Vec and Glove [4]. The FastText algorithm treats n-grams at the character level of a word as the smallest unit. Using special delimiter symbols at the beginning and end, words are divided into subwords of length n . For example, the FastText representation of the word “detay” is $\text{;de,det,eta,tay,ay;}$ if the n -gram is equal to 3. After the composition of these n -gram subwords, the embedding vector of the word is calculated as the average of these n -gram vectors. The remaining word learning processes are similar to those of Word2Vec, where the neighboring words of the context are learned using the skip-gram approach. FastText also has a text classification module that uses a neural network to predict the class labels of input documents. The mean values of the word embedding vectors form the document vector for the neural network model. After the model is trained, documents or any text sentences can be classified using the model.

2.6. ITU NLP Tool

One of the main challenges in natural language processing studies is that each language has its own unique linguistic structure. For example, in agglutinative languages, word stems often change and become unrecognizable when new words are derived, while in polysynthetic languages, words are conjugated by adding prefixes/suffixes that do not cause stem changes [6]. Therefore, natural language processing algorithms should be developed in a language-specific manner to improve performance. ITU NLP Tool (Istanbul Technical University Natural Language Processing Tool) is a web-based service specified for Turkish languages and includes various linguistic analysis modules such as Tokenizer, Normalizer, Morphological Analyzer, etc. [12]. The application analyzes and parses the given text with different modules and uncovers linguistic word components in the text, such as word stems, conjunctions, adverbs and adjectives. To clean and parse the words in the dataset, we used the tool ITU NLP in our experiments.

2.7. Cosine Similarity

To perform a comparison of vectors, we need a measure that can calculate the similarity of given vectors. Cosine similarity is one of the well-known methods to determine the similarity between vectors that are not zero in an inner product space. Mathematically, it measures the cosine of the angle between two vectors projected into a multidimensional space, and the result is clearly limited to $[0,1]$.

$$\cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|} = \frac{\sum_1^n a_i b_i}{\sqrt{\sum_1^n a_i^2} \sqrt{\sum_1^n b_i^2}}$$

The cosine similarity of two given vectors \mathbf{a} and \mathbf{b} is calculated by formula (1), where $\mathbf{a} \cdot \mathbf{b}$ is the dot product of the vectors and $\|\mathbf{a}\|$, $\|\mathbf{b}\|$ are the length of the two vectors. If the result of $\cos\theta$ is equal to 1, it means that the given vectors are exactly equal [18].

3. Proposed Approach

Our system is designed to predict the most suitable staff when a task request is received through the portal system. Assigning a task that requires specific qualifications to a staff member who lacks adequate experience in those qualifications may result in delays, non-completion, and financial losses. To address this issue, it is crucial to assign the most qualified personnel to tasks. However, this approach may lead to imbalances in job-sharing among staff and conflicts arising from simultaneous tasks across different projects. Therefore, the system should offer multiple staff members, sorted based on their ability to meet the task requirements. For instance, if a web development task requires design alterations, a staff member specializing in front-end development could be sufficient. However, for the development of a complete website, it is more efficient in terms of cost to assign a staff member with experience in both back-end and front-end development. By considering the specific requirements of each task, our system aims to optimize staff allocation and ensure task completion in a timely and cost-effective manner. In order to ensure that our system compares needs of task and staff-wise qualification.

The system comprises two primary steps to fulfill its functionality. Firstly, the task text is processed to identify the required skills, such as development, help desk, RD (Research and Development). The task management module serves as a repository, storing comprehensive information including task descriptions, subjects, and project names. This module integrates with the company's ERP systems and forwards the received tasks to the text mining module through the "Task Repository," which acts as the database for the management module.

Text Mining module performs to text data a series of operations such as the removal of tags and punctuation. These operations ensure data cleanliness of the text data. We also tried different types of pre-processing using the text mining methods such as stop-words removal, word normalization and stemming as described in the dataset section. The other key point has effect on performance is vectorization of the text data. Vectorization is converting process of a data type to vectors that enables to analyses of textual data with machine learning. Within our study, we experimented with 4 different vectorization methods, namely TF, TFIDF, Word2Vec and Doc2Vec to compare the effect of different vectorization methods and select the best method for the live system. The Python library Scikit-Learn was used to implement the TF and TFIDF vectorization approaches [25]. These methods do not require a learning algorithm since they vectorize the words based on the text statistics.

The Word2Vec and Doc2Vec methods are essentially a neural network model with an input, a hidden, and an output layer that attempts to uncover the semantic relationship between words. It is necessary to train these models to learn the semantic similarities specific to the data. However, it is crucial to avoid utilizing test data during the optimization

process to prevent overfitting and ensure fair predictive performance of the model. Otherwise, it is concluded that the results to be obtained are not reliable. Hence, the training dataset is re-split into 80% validation training and 20% validation testing. Figure 3 summarizes all the steps of our proposed system.

The second main step of proposed system involves identifying the optimal staff member for a classified task. The Staff Qualification module, which contains personnel information and the up-to-date qualifications, is used when deciding by system on the assignment of new tasks to an employee who is the most suitable. The system utilizes 11 valued vectors, corresponding to class distribution, to represent the qualifications of individual staff members. After receiving the predicted task text vector from the classification modules and the personnel qualification vectors from the Staff Qualification module, the Decision Support System is activated to compare these vectors. In our approach, we employed cosine similarity for this comparison. Cosine similarity allows for a value-wise comparison between vectors, where each value corresponds to a specific label. For example, value corresponding to "Abap" label is compared to "Abap" value of prediction. However, total similarity is calculated by summing the element-wise similarities. For instance, if a task involves full web development along with some data mining analysis, our machine learning model possibly predict a vectors with higher values for "ReDe" and "Hybr" classes. Because data mining and web developments are concerned with "ReDe" and "Hybr" classes, respectively. Therefore decision support module should recommend some staffs who have experience for both of classes. Cosine similarity will calculate higher score to a staff member whose qualification vector has higher values for both "ReDe" and "Hybr" compared to a staff member who only excels in "ReDe" but has lower values for "Hybr". This approach ensures that our model predominantly recommends the most suitable staff members, thereby optimizing resource allocation and reducing costs. Once a task is assigned to the staff by the system, upon its completion, the personnel vector is updated based on the predicted task text vector obtained from the machine learning model. As personnel specializing in one or more areas successfully accomplish tasks, their qualifications are also updated, enabling the company to monitor and track the continuous improvement of its personnel.

4. Experiment Results

Our live system runs with initialization of modules, prediction of related fields and estimation of the most suitable staff to the user for a newly received task from projects of the consulting company. To obtain a reliable and stable estimation, the system needs to predict the field for a given task text as accurately as possible, and this is only possible if the machine learning models have high predictive performance. Therefore, in an experimental step, different approaches for text vectorization, machine learning, and preprocessing were compared.

The first stage of experimental work includes data organization and preprocessing steps. The process of generating datasets involves four main steps. The first step is to collect and tag project/task requests. The request tickets contain real-time project/task requests from Detaysoft [1] which is one of the largest SAP consulting companies in Turkey. The company uses its portal system to manage tasks, resources and business. Requests are created and submitted by clients, managers or team leaders through the portal, along with

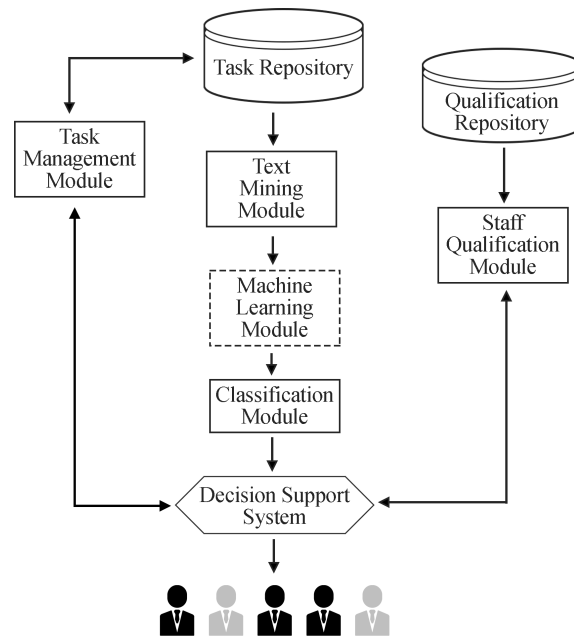


Fig. 3. Architecture of end-to-end Decision Support System

the selection of employees to be assigned the tasks. The collected request tickets are then downloaded, and irrelevant parts such as XML or HTML tags are removed. The result is a dataset with 2103 texts explaining tasks.

The second step is to label the relevant data. Since the company does not have an existing labeling pool, a qualitative research method called grounded theory is used to establish the most accurate labeling pool. Project managers and team leaders processed the raw data according to predefined rules and defined 11 class labels to label the samples. These classes represent SAP installation, update and configuration processes "Bsis," software development language "Abap," human resources processes "HrGn," e-commerce platform operations "Hybr," and data processing and reporting processes "BwBo." Logistics processes are represented by three subclasses: Sales-Distribution "LoSd," Quality Management "LoQm" and Materials Management "LoMm" Financial transactions are represented by the labels "FiCo" for cost accounting and "FiRe" for financial property management. The label "ReDe" is assigned to jobs that fall outside these 10 defined classes and usually require special expertise.

The third stage is the assignment of employee qualifications based on 11 predefined class labels by Grounded Theory. Machine learning models were also used to perform the assignment process systematically and based on the tasks performed by each employee. In this model, the employee qualifications dataset labeled by the team leaders was used. With this model, it is aimed to predict in which areas a staff whose competencies are given in writing is prone to develop projects. Since the team leaders will make improvements in the data labeling part, this model used in data labeling has only been used to speed up the data labeling process. In this model, Word2Vec and LR were used and

79% accuracy was achieved. For every completed task, a trained machine learning model predicts a qualification vector comprising the 11 class distributions, which is then set as the qualification distribution of the respective staff member. This approach enables the measurement of staff experience and qualifications in different fields. However, since the prediction performance of machine learning models is not 100%, the skill vectors were set as changeable by the team leaders or the employees themselves against wrong values.

The last step involves the generation of different datasets through various preprocessing techniques. In the text mining literature, it is well known that the use of unnecessary words, word normalization, and stemming have positive or negative effects on model success [32]. To show the impact of text preprocessing on machine learning prediction and overall system success, the raw data was preprocessed by removing stop words, normalizing words, and stemming words in ITU NLP tool, and then stored separately for the experiment (see Figure 4).

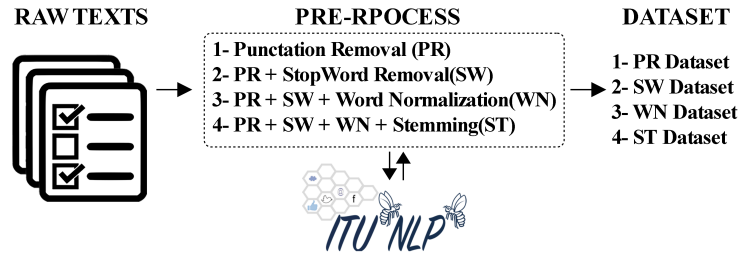


Fig. 4. Creation Steps of Different Dataset based on pre-processing

After pre-processing steps, dataset was divided into two parts to train and to test proposed model. The numerical distribution of the data used for the training and testing process in the models is shown in Figure 5. The data size of the training/testing sets was balanced to avoid class-specific overfitting and to ensure the model learned for all classes. Performance metrics are calculated by comparing the actual label of the task text in the dataset with the predicted class label. For instance, if a test task belongs to the "ReDe" class and our model predicts it as "ReDe", it is considered a truly predicted sample. However, as previously mentioned, our model predicts an 11-valued vector prediction instead of a single class. To provide a fair assessment of the vector-based predictions, we also calculated the AUC score, which takes into account the overall performance of the vector values. This allows for a more comprehensive evaluation and comparison of the model's predictions.

In the next phase, features were extracted using TF, TFIDF, Word2Vec and Doc2Vec. As mentioned before, TF and TFIDF methods do not need training, but Word2Vec and Doc2Vec methods need a training process as they are artificial neural network models. In order to train Word2Vec and Doc2Vec models and find optimal hyperparameters the GENSIM[28] and Optuna [2] libraries was used, respectively. Detailed information about the parameter spaces for these two models can be found in Table 1. As mentioned earlier, the main purpose of distributed word representation methods is to determine the vectors of words based on their semantic similarities. However, our goal in this study is to classify

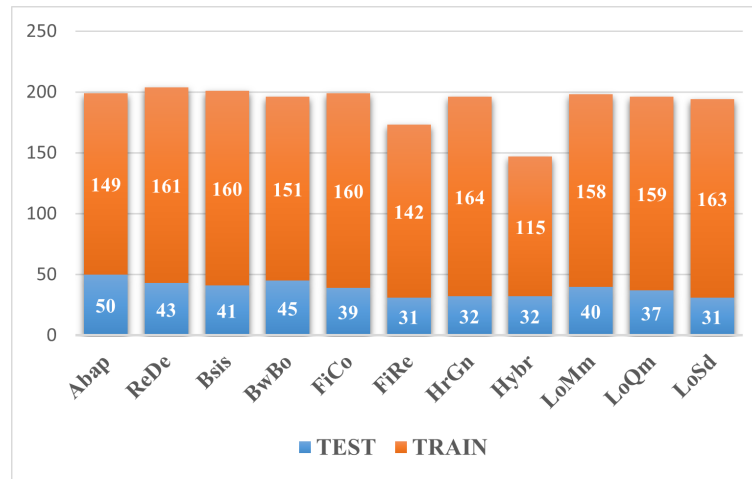


Fig. 5. Numerical distribution of samples in training and testing phase

the text of a task sample that contains more than one word, and even sentences. Therefore, it is necessary to calculate the vector of the sample from word vectors. To overcome this challenge, it is chosen to use word vectors learned using word embedding methods, and the vector of each task sample was formed by averaging the word vectors it contains. A random set of parameters from the given space is automatically set in the model for each iteration to perform a training and testing process in the optuna library. Therefore, the validation data for the hyper-parameter optimization of Word2Vec and Doc2Vec are reformed after each iteration depending on the parameter group.

Table 1. Parameter Space of Word2Vec and Doc2Vec representation methods for hyper-parameter optimization

Parameter	Parameter Space
Vector Size, Dimensionality of the word vectors	(5 - 5000)
Word Window, distance between current predicted words	(2 - 10)
Alpha, The initial learning rate	(0.00001, 10)
Epoch, number of iteration over dataset	[50,100,250,500,750,1000]
Training algorithm: 1 for skip-gram; otherwise CBOW.	[0,1]
Minimum Count, Min. Number of occurrence of words.	(1 - 10)

Following the vectorization procedures applied to the raw text samples, the datasets were prepared for utilization in machine learning models. Table 2 provides an overview of the vector lengths for each sample across various vectorization approaches. Notably, the Word2Vec and Doc2Vec-based vectors exhibit slightly shorter lengths due to the nature of these methods, which involve embedding word representations.

Table 2. Vector lengths of samples after each pre-process methods

Pre-Process Method	TF	TFIDF	W2V	D2V
Word Normalized Data	12969	12969	4178	2227
Word Stemmed Data	5554	5554	1546	4309
Stop Words Removed Data	13708	13708	4995	812
Punctuation Removed Data	13919	13919	2899	4891

In addition to pre-processing methods, choosing the most successful classification algorithm for the live application system is also crucial for assigning tasks to the optimal personnel. Hence, our study involved a comparison of five different machine learning algorithms: logistic regression (LR), support vector machines (SVM), random forest (RF), k-nearest neighbor (KNN), XGBoost (XGB), and FASTText. Each dataset generated using various approaches was trained and tested with these algorithms except FASTText. Because, FastText classification module automatically converts the texts to vectors using its own Word2Vec method. Therefore, no other vectorized datasets were used for FastText, only preprocessed data. Subsequently, the best-performing model was selected and integrated into the end-to-end system. We also optimized hyperparameters due to reason of algorithms has a significant impact on predictive performance. To make our system more robust, the hyperparameters of the different algorithms were optimized separately using the Optuna library. The space and list of hyperparameters of each algorithm that were optimized are shown in Table 3.

Table 3. Parameter Space of machine learning algorithms for hyper-parameter optimization

Classification Algorithm	Parameter	Parameter Space
LR	Regularization (C)	$2^{-12} - 2^{12}$
	Solver Algorithm	Linear, BFGS
SVM	Regularization (C)	$2^{-12} - 2^{12}$
	Kernel	Linear, RBF, Polynomial
	Gamma	(0.000001 - 10)
RF	Number of Trees	(2 - 1000)
	Criterion	Gini, Entropy
KNN	Number of Neighbors	(2 - Number Of Samples)
	Algorithm	KD-Tree, Ball-Tree, Brute
XGB	Number of Trees	(2 - 1000)
	Learning Rate	(0.0001 - 10)
	Alpha	(0 - 32)
	Gamma	(0 - 32)
FASTTEXT	Vector Size	(3 - 5000)
	Window Size	(1-15)
	Length of Word n-gram	(1-5)
	Training Algorithm	[CBOW, Skip Gram]
	Epoch	[50,100,250,500,750,1000]
	Learning Rate	(0.00001 - 1)

Table 5 shows the results of the machine learning models for "raw data," i.e., no pre-processing was done except for tag removal. It can be seen that the TF-IDF vectorization method achieves the best results for all algorithms when analyzing the "raw data". The Word2Vec method, on the other hand, seems to be the most unsuccessful representation method for "raw data". Since the word embedding algorithms focus on the neighborhood relationship of words in the sentence, the models may have learned the relationship between stop words rather than between keywords, and therefore may not have predicted the correct vector representation that contains characteristic features related to classification into the correct class.

Table 5. Accuracy and AUC scores of machine learning algorithms on Raw Dataset which no any pre-processed applied

Method	Accuracy				AUC			
	TF	TFIDF	W2V	D2V	TF	TFIDF	W2V	D2V
LR	0.7625	0.7886	0.5843	0.7648	0.9594	0.9697	0.9058	0.9611
SVM	0.7458	0.7981	0.5487	0.715	0.9578	0.9721	0.8941	0.9563
KNN	0.1639	0.6746	0.4774	0.6437	0.8072	0.9452	0.87	0.9226
RF	0.7268	0.715	0.5938	0.6603	0.9549	0.9506	0.9106	0.9366
XGB	0.6817	0.6461	0.5677	0.5582	0.9339	0.9206	0.9063	0.8824

The Word2Vec models achieved the best predictive performance among all methods and models, with an accuracy of 80.52% in the dataset containing no stopwords (see Table 6). This result also proves that stop words have the opposite effect of the word embedding methods in terms of prediction performance, as explained previously. 80.52% accuracy obtained using real-time data with our proposed method is a similar accuracy with the models developed using historical data in the literature. Obtaining similar accuracy with real-time data caused us to evaluate our model as successful. In addition, the study proposes a semi-autonomous system and leaves the final choice to the decision makers. For this reason, it is predicted that model error rates close to 20% can be corrected by decision makers. Our model will automate the pre-assignment tasks that require a lot of work, thus allowing decision makers to exert much less effort.

Table 6. Accuracy and AUC scores of machine learning algorithms on Stop-word cleaning applied Dataset

Method	Accuracy				AUC			
	TF	TFIDF	W2V	D2V	TF	TFIDF	W2V	D2V
LR	0.7648	0.7933	0.7791	0.734	0.959	0.9717	0.9713	0.9573
SVM	0.7316	0.7933	0.8052	0.7055	0.959	0.9735	0.9749	0.9549
KNN	0.2375	0.6627	0.7173	0.4537	0.6613	0.945	0.9441	0.8313
RF	0.7221	0.7221	0.7791	0.7245	0.9529	0.9514	0.966	0.9437
XGB	0.0974	0.5796	0.7791	0.4608	0.5	0.8973	0.9701	0.8655

The results of the data set containing normalization and stemming preprocessing are shown in tables 7 and 8, respectively. There were no significant changes in the performance of the machine learning algorithms for either. However, some results show extremely low performance, such as the 38% accuracy of KNN in the stemmed dataset or the 31% accuracy of XGB in the normalized dataset. These low accuracies can be explained by overfitting due to the large hyperparameter optimization space of the algorithms.

Table 7. Accuracy and AUC scores of machine learning algorithms on normalization process applied Dataset

Method	Accuracy				AUC			
	TF	TFIDF	W2V	D2V	TF	TFIDF	W2V	D2V
LR	0.772	0.7838	0.7363	0.6746	0.9626	0.9718	0.9627	0.9436
SVM	0.7411	0.7815	0.7862	0.6437	0.9604	0.9751	0.9657	0.9263
KNN	0.1591	0.677	0.6722	0.5701	0.8601	0.9489	0.9456	0.9171
RF	0.7245	0.7292	0.772	0.6698	0.9541	0.9529	0.9673	0.942
XGB	0.3135	0.5202	0.7221	0.6461	0.7071	0.86	0.9638	0.9362

Table 8. Accuracy and AUC scores of machine learning algorithms stemming process applied Dataset

Method	Accuracy				AUC			
	TF	TFIDF	W2V	D2V	TF	TFIDF	W2V	D2V
LR	0.7458	0.7815	0.7292	0.7197	0.9519	0.971	0.9587	0.9513
SVM	0.7458	0.791	0.7672	0.696	0.9555	0.974	0.9665	0.9397
KNN	0.38	0.6556	0.7126	0.6057	0.7539	0.9463	0.9234	0.9008
RF	0.7672	0.7316	0.7815	0.715	0.9673	0.9642	0.9616	0.9477
XGB	0.5653	0.5629	0.7387	0.715	0.8912	0.9231	0.9609	0.9577

FASTText, another algorithm applied to our dataset, failed to outperform the other machine learning methods, achieving a low prediction accuracy of 67%. The algorithm splits each word within the sample text based on a given n-gram size for representation; therefore, the word vectors may have lost their discriminative power with respect to the field labels.

For example, the FASTText representation of the word "detay" contains $\langle "de", "et", "ta", "ay" \rangle$ sub-word vectors when the n-gram size is set to 2. But the word "ay" also has many different meanings in Turkish. Therefore, it may be possible that contribution of sub-words to representation has some negative effect in Turkish language.

The Receiver Operating Characteristic (ROC) curves for each class obtained from the SVM-Word2Vec-NOSW model, which performed the best among all models, are shown in Figure 7. The ROC curve is one of the performance measurement methods that shows how well the model can separate classes for a binary problem. The ROC curve is plotted with the TPR (True Positive Rate) against the FPR (False Positive Rate), with the TPR on

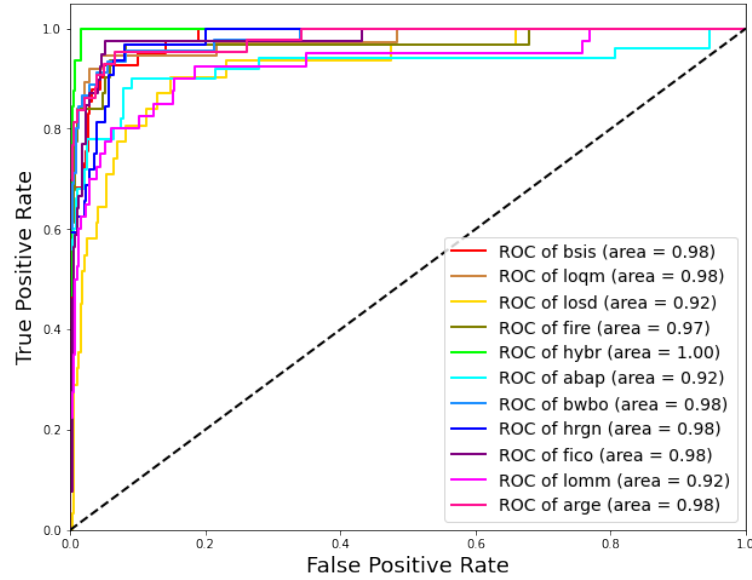


Fig. 7. Plot of Area Under the Curve (AUC) for each class

the y-axis and the FPR on the x-axis. If the AUC value is equal to 1, this indicates that the model is excellent at discriminating between the class labels of the samples. Since the dataset used in our study consists of 11 classes, the OvR (One-versus-Rest) approach was used, where each class was left alone to label the problem as binary to plot the ROC curves.

After predicting the class vectors of the new task text, the recommendation process is started. Our system was developed as a microservice that can be integrated into a company's portal or ERP system. The front-end design of our application is shown in Figure 8. When a task is created and the information fields such as subject, description, project and deadline are filled in, the automatic system is triggered to recommend the most suitable candidates for the task. Although the machine learning model predicted the text as "LoSd" class, our model recommended some employees who have skills for "LoQm" first. The reason for this result is that our machine learning model does not predict a specific class, but a vector of class distribution. It can be interpreted that the vector of task input might have included some different requests for "LoSd" besides "LoQm", and therefore employees who have "LoQm" skills are ranked first because the decision support system recommends the most suitable candidate according to cosine similarity.

5. Conclusion

Efficiently managing multiple projects and ensuring timely completion poses significant challenges for IT and consulting firms. It is necessary to assign tasks to the right people in order to complete the tasks at hand together with limited human resources. The assignment of tasks can only be decided by senior staff such as project managers or RD

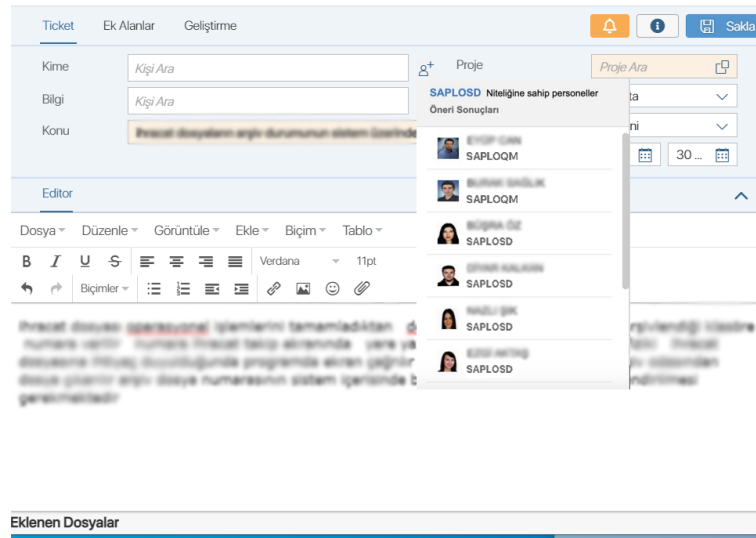


Fig. 8. User interface of proposed recommendation system

managers who know the qualifications of the staff. However, managing and controlling task assignment is somewhat tedious, especially in large organizations that have numerous teams and employees with different skills and qualifications.

Implementing an automated system that processes tasks and quickly recommend the most suitable employees has the advantage of shortening project run times and avoiding potential assignment problems. Our work involves the development of a decision support service to recommend the most appropriate employees by analyzing the task text using various methods to facilitate task assignment.

When a new task is entered through the system, the vector containing the task requirements is automatically predicted and employees are recommended based on skill similarities. Once the given tasks are completed, the qualifications of the assigned personnel are automatically updated with the task vector, using the same system. In this way, tracking the progress of personnel is simplified and does not require further human intervention. Since each employee is familiar with the required tasks and the qualifications are updated regularly, the distribution of qualified personnel within the company can be systematically and statistically recorded to support the HR department.

Establishing an automatic system, which handles tasks and quickly recommend the most suitable employees, provide advantages on shorten completion times of project and might have prevent potential assignment issues. Our work includes the development of the decision support service, which aims to recommend the most suitable personnel by analyzing the task text with various methods in order to facilitate task assignments. Unlike other studies in the literature, our system developed to integrate a living real-time system rather than the perform an optimization on static task/personnel dataset. When a new task is entered over the system, the vector which includes requirement of task is automatically predicted and employee are recommended based on qualification similarities. Just after the related tasks is completed, qualifications of assigned personnel is updated with task

vector automatically using the same system. Thus, the tracking of personnel progress is simplified and not need more human interference. Besides, since the each employee get experienced with skill required tasks and qualifications are updated regularly, the qualified staff distribution within the company can reported in systematically and statistically to use in process of human resource department.

Since the main goal of our study is to present the general architecture of a decision support system working with integrated enterprise software, some new generation transformer-based text representation approaches such as ELMo [26], BERT [10] have not yet been considered. In future studies, the success rate of the recommender system will be improved by adding new class labels for subdepartments and using state-of-the-art text representation models. In addition, although employee workload is recorded in the portal system, this information is not functionally used in task assignment. In the next version of the application, a more complex system will be developed that includes workload, qualifications, and task assignments.

References

1. SAP Global | Platin İş Ortağı - Detaysoft, <https://detaysoft.com/tr-TR/index>
2. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2623–2631 (2019)
3. Al-Hawari, F., Barham, H.: A machine learning based help desk system for IT service management. *Journal of King Saud University-Computer and Information Sciences* 33(6), 702–718 (2021), ISBN: 1319-1578 Publisher: Elsevier
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching Word Vectors with Subword Information. Tech. Rep. arXiv:1607.04606, arXiv (Jun 2017), <http://arxiv.org/abs/1607.04606>, arXiv:1607.04606 [cs] type: article
5. Cai, X., Li, K.N.: A genetic algorithm for scheduling staff of mixed skills under multi-criteria. *European Journal of Operational Research* 125(2), 359–369 (2000), ISBN: 0377-2217 Publisher: Elsevier
6. Carki, K., Geutner, P., Schultz, T.: Turkish LVCSR: towards better speech recognition for agglutinative languages. In: 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100). vol. 3, pp. 1563–1566 vol.3 (Jun 2000), ISSN: 1520-6149
7. Charmaz, K., Belgrave, L.L.: Thinking about data with grounded theory. *Qualitative Inquiry* 25(8), 743–753 (2019), ISBN: 1077-8004 Publisher: SAGE Publications Sage CA: Los Angeles, CA
8. Chen, R., Liang, C., Gu, D., Leung, J.Y.: A multi-objective model for multi-project scheduling and multi-skilled staff assignment for IT product development considering competency evolution. *International Journal of Production Research* 55(21), 6207–6234 (2017), ISBN: 0020-7543 Publisher: Taylor & Francis
9. Cheng, H., Chu, X.: Task assignment with multiskilled employees and multiple modes for product development projects. *The International Journal of Advanced Manufacturing Technology* 61(1), 391–403 (2012), ISBN: 1433-3015 Publisher: Springer
10. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Tech. Rep. arXiv:1810.04805, arXiv (May 2019), <http://arxiv.org/abs/1810.04805>, arXiv:1810.04805 [cs] type: article
11. Dooley, L., Lupton, G., O’Sullivan, D.: Multiple project management: a modern competitive necessity. *Journal of Manufacturing Technology Management* 16(5), 466–482 (Jan 2005),

- <https://doi.org/10.1108/17410380510600464>, publisher: Emerald Group Publishing Limited
12. Eryiğit, G.: ITU Turkish NLP web service. In: Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics. pp. 1–4 (2014)
 13. Garcia, I., Pacheco, C., Arcilla-Cobián, M., Calvo-Manzano, J.: Mympm: A plug-in for implementing the metamodeling approach for project management in small-sized software enterprises. *Computer Science and Information Systems* 13(3), 827–847 (2016)
 14. Guo, L., Vargo, C.J., Pan, Z., Ding, W., Ishwar, P.: Big social data analytics in journalism and mass communication: Comparing dictionary-based text analysis and unsupervised topic modeling. *Journalism & Mass Communication Quarterly* 93(2), 332–359 (2016)
 15. Hartmann, S., Briskorn, D.: A survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of operational research* 207(1), 1–14 (2010), ISBN: 0377-2217 Publisher: Elsevier
 16. Kane, H., Tissier, A.: A Resources Allocation Model for Multi-Project Management. In: 9th International Conference on Modeling, Optimization & Simulation (2012)
 17. Lagesse, B.: A Game-Theoretical model for task assignment in project management. In: 2006 IEEE International Conference on Management of Innovation and Technology. vol. 2, pp. 678–680. IEEE (2006)
 18. Lahitani, A.R., Permanasari, A.E., Setiawan, N.A.: Cosine similarity to determine similarity measure: Study case in online essay assessment. In: 2016 4th International Conference on Cyber and IT Service Management. pp. 1–6 (Apr 2016)
 19. Le, Q.V., Mikolov, T.: Distributed Representations of Sentences and Documents. Tech. Rep. arXiv:1405.4053, arXiv (May 2014), <http://arxiv.org/abs/1405.4053>, arXiv:1405.4053 [cs] type: article
 20. Li, X., Nie, M., Yang, G., Wang, X.: The study of multi-project resource management method suitable for research institutes from application perspective. *Procedia Engineering* 174, 155–160 (2017)
 21. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
 22. Mitkov, R.: *The Oxford handbook of computational linguistics*. Oxford University Press (2004)
 23. Mo, Y., Zhao, D., Du, J., Syal, M., Aziz, A., Li, H.: Automated staff assignment for building maintenance using natural language processing. *Automation in Construction* 113, 103150 (2020), ISBN: 0926-5805 Publisher: Elsevier
 24. Möhring, R.H.: Minimizing costs of resource requirements in project networks subject to a fixed completion time. *Operations Research* 32(1), 89–120 (1984), ISBN: 0030-364X Publisher: INFORMS
 25. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V.: Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12, 2825–2830 (2011), ISBN: 1532-4435 Publisher: JMLR. org
 26. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. Tech. Rep. arXiv:1802.05365, arXiv (Mar 2018), <http://arxiv.org/abs/1802.05365>, arXiv:1802.05365 [cs] type: article
 27. Ponstee, A., Kusters, R.J.: Classification of human-and automated resource allocation approaches in multi-project management. *Procedia-Social and Behavioral Sciences* 194, 165–173 (2015), ISBN: 1877-0428 Publisher: Elsevier
 28. Rehurek, R., Sojka, P.: Gensim–python framework for vector space modelling. NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic 3(2), 2 (2011)
 29. Shahariar, G.M., Biswas, S., Omar, F., Shah, F.M., Hassan, S.B.: Spam review detection using deep learning. In: 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). pp. 0027–0033. IEEE (2019)

30. Vijayarani, S., Ilamathi, M.J., Nithya, M., et al.: Preprocessing techniques for text mining-an overview. *International Journal of Computer Science & Communication Networks* 5(1), 7–16 (2015)
31. Vogrinčič, S., Bosnić, Z.: Ontology-based multi-label classification of economic articles. *Computer Science and Information Systems* 8(1), 101–119 (2011)
32. Vyas, M.J., Bhandari, S.D.: A Survey on Pre-processing Techniques for Text Mining. *Data Mining and Knowledge Engineering* 6(2) (2014), <http://www.ciitresearch.org/dl/index.php/dmke/article/view/DMKE022014006>, number: 2

Halil Arslan received BS, MS and, PhD, degree in electronic and computer education from Sakarya University, Turkey in 2004–2015 respectively. Since 2017, he has been teaching operation systems, cyber security and, computer networks courses as an assistant professor in the Computer Engineering Department at University of Sivas Cumhuriyet. His research interests are computer networks, cyber security and, software engineering.

Yunus Emre Işık received his bachelor's and master's degrees in Management Information Systems from Mehmet Akif Ersoy University and Cumhuriyet University, respectively. He is pursuing his Ph.D. in Electrical and Computer Engineering at Abdullah Gul University. He is currently working as a research assistant in the Department of Management Information Systems at Cumhuriyet University in Sivas, Turkey. His research focuses on the implementation of AI models in various domains such as text mining, image processing and bioinformatics.

Yasin Görmez received the graduate degree from Computer Engineering Department, Meliksah University, the MSc degrees with high honor from the Electrical and Computer Engineering Department, Abdullah Gul University, and Ph. D. degrees with high honor from the Electrical and Computer Engineering Department, Abdullah Gul University, in 2015, 2017 and 2022 respectively. He served as a research assistant between 2015 and 2023 in management information systems department of Sivas Cumhuriyet University, Sivas, Turkey. Now, he is an assistant professor in management information systems department of Sivas Cumhuriyet University, Sivas, Turkey. He has particularly honed his skills in designing deep learning methods and has developed deep learning techniques in various fields such as bioinformatics, natural language processing, image processing, and cybersecurity

Mustafa Temiz graduated from Computer Engineering Department of Erciyes University. He received the master's degree in management information systems from Sivas Cumhuriyet University. Currently a Ph.D. student in Electrical and Computer Engineering at the Abdullah Gul University and a research assistant in Department of Management Information Systems, Cumhuriyet University, Sivas, Turkey.

Received: September 22, 2023; Accepted: October 11, 2023.

Activity Recognition for Elderly Care Using Genetic Search

Ankita Biswal¹, Chhabi Rani Panigrahi^{2,*}, Anukampa Behera³, Sarmistha Nanda⁴,
Tien-Hsiung Weng^{5,*}, Bibudhendu Pati², and Chandan Malu⁶

¹Dept. of Computer Science & Engineering, CUTM, Bhubaneswar, India
ankitabiswal94371@gmail.com

²Department of Computer Science, Rama Devi Women's University, Bhubaneswar, India
{panigrahichhabi, patibibudhendu}@gmail.com

³Department of Computer Science & Engineering, S'O'A Deemed to be University,
Bhubaneswar, India
anukampal@gmail.com

⁴Department of Computer Science & Engineering, Gandhi Engineering College,
Bhubaneswar, India
sarmisthananda@gmail.com

⁵Department of Computer Science and Information Engineering, Providence University,
Taichung 43301, Taiwan
thweng@gm.pu.edu.tw

⁶iCETS, Infosys, Bhubaneswar, India
chandanmalu@gmail.com

Abstract. The advent of newer and better technologies has made Human Activity Recognition (HAR) highly essential in our daily lives. HAR is a classification problem where the activity of humans is classified by analyzing the data collected from various sources like sensors, cameras etc. for a period of time. In this work, we have proposed a model for activity recognition which will provide a substructure for the assisted living environment. We used a genetic search based feature selection for the management of the voluminous data generated from various embedded sensors such as accelerometer, gyroscope, etc. We evaluated the proposed model on a sensor-based dataset - Human Activities and Postural Transitions Recognition (HAPT) which is publically available. The proposed model yields an accuracy of 97.04% and is better as compared to the other existing classification algorithms on the basis of several considered evaluation metrics. In this paper, we have also presented a cloud based edge computing architecture for the deployment of the proposed model which will ensure faster and uninterrupted assisted living environment.

Keywords: Activity Recognition; HAR; Genetic Search Algorithm; HAPT; SMO; Edge Computing; Cloud Computing.

1. Introduction

Old age refers to the critical part of life where a person is more prone to various accidents and life threatening hazards. As per the survey led by World Health Organization (WHO), old age population will grow by 56%, from 962 million in the year 2017 to 1.4 billion speculated in 2030 and the population will be doubled to 2.1 billion by the year 2050 [2]. Old age requires proper care, comfort as well as alert monitoring but today's fast-paced lifestyle of family members do not allow them to always stay near and be vigilant. So it is the need of the hour to provide assisted living based on activity recognition. HAR is considered as evolutionary technology listed under the category of pervasive computing which is applicable to lots of real-life-care activities such as assisted living, smart home models, disaster detection, healthcare, fitness tracking, etc. [1]. This assisted living with healthcare monitoring facility can be extended to elderly people without invading their privacy which is a very important aspect. This concern has been addressed with the use of implanting sensors to various wearable gadgets. Now for monitoring and analysis purpose, the HAR is generally obtained by collecting, compiling, and analyzing sensor or visual data [4]. Apart from this for research purpose, many sensor-based datasets such as OPPORTUNITY, MHEALTH, WISHDM, UCI-HAR [7], HAPT [8], etc. are also available in the public domains which are obtained by collecting data from both body-worn sensors and ambient sensors. There are some activity recognitions data collected from subjects in natural out-of-lab settings but do not quantify recognition accuracy [6].

Over last two-decade various research works have been done in the area of HAR. Starting from Multi-Layer Perceptron (MLP), Support Vector Machine (SVM), and Radial Basis Function (RBF) to knowledge based approach, Convolutional Neural Network (CNN) based models [37, 38, 39], Dual-Path CNN-RNN models are used to classify activities [31]. The dire necessity is to find a method to classify the activities that minimizes the cost involved while maximizing the production efficiency [5, 40]. In this regard the use of smart phones embedded with built-in sensors such as microphones, dual cameras, accelerometer, gyroscopes, etc. can be considered as a pertinent option to enable cost minimization objective.

1.1. Motivation

After studying the various works done in the field of HAR, we could find the following limitations:

- The use of vision-based activity recognition hampers the privacy of elderly people.
- Some authors use environmental sensors which defy portability to achieve mobility, and also might give trouble if multiple people are in the same space.
- The use of wearable sensors might be uncomfortable and irritable for elderly persons.
- Scarcity of data in ontological, knowledge-driven method fails to define abnormality in the data.
- A variant of accuracies and time complexities were found by using different classification techniques. We can say that the quest to achieve higher accuracy gives

rise to higher time complexity [27] which can be a big problem in the implementation of Mobile-based health application.

In all the works done in the literature, basically two challenges arise those have motivated us towards our proposed work such as:

As data acquisition through these smart phones contains a large number of features; it makes classification susceptible to over-fitting and increases the training and testing time of classification [18]. To avoid this, feature selection contributes a lot in getting the essential features as a subset of original features to avoid over-fitting. It can be filter-based, wrapper-based, and embedded. In the case of elderly activity recognition, filter-based feature selection is preferred as it preserves the original features and finds a subset of best features by keeping accuracy as a threshold [36]. Even though, the classification algorithms those have their primary potential as features selection are used in embedded feature selection, the demand for resource usage is huge when computed [9]. After dimensionality reduction, the next task performed is activity classification for which many Machine Learning based algorithms have been used.

Utilizing wearable gadget's limited resources such as memory, energy, embedded sensors may lead to a decrease in computation performance and efficiency of the system. It is also important to develop a model which is movable, consumes low-battery, uses low-cost sensors, gives high accuracy, and preserves the privacy of the user [11]. So the amalgamation of Edge computing with Mobile Cloud computing can be a feasible solution.

1.2. Contributions

Keeping into consideration the above mentioned limitations and challenges; we propose a model that has a threefold contribution:

- We have proposed a cost efficient feature selection based approach for HAR classification.
- A comparison of obtained results of activity recognition with and without implementation of feature selection was performed.
- We have also proposed a cloud-based architecture for the development of a mobile application of HAR for elderly care.

Section 2 presents a summarization of related works done in this area with the limitations The proposed methodology is discussed in Section 3 and the experimentation with results is stated in Section 4. In Section 5, proposed cloud architecture is described and Section 6 presents the conclusion and future work.

2. Related Work

HAR has a crucial functionality towards the management of several health risks and diseases and providing an assisting environment is the purpose of an activity recognition system. So many works have been proposed by several researchers regarding activity recognition in the last two decades. In this section, we have discussed the major works

done in the field of HAR to develop elderly care applications along with the corresponding limitations and research gaps.

Most recently, a hybrid LSTM was implemented with the Bayesian optimizer which achieved 99.39% of accuracy when experimented with the UCI-HAR dataset [28], but the model does not contain any postural transitions (sit-to-stand, stand-to-sit, sit-to-lie, etc.). A CNN-LSTM model integrated with Adam optimizer, when tested for HAR with datasets WISDM and UCI-HAR, obtained an accuracy of 95.85% and 95.78% respectively [27], but the computation is mentioned to be slow as the dependency in between does not allow parallel processing. Researchers have also used the Sequential Floating Forward Search (SFFS) feature extraction technique and SVM classification algorithm and achieved 96.81% accuracy. Using deep embedding derived from a fully convolutional neural network to classify the activities of several datasets and the WISDM dataset gave an accuracy of 91.3% with Personalized Triplet Network (PTN) [3]. A binary classification dataset has been proposed to achieved an accuracy of 96.81% by applying Random Forest [26].

MLP which implements Xavier's algorithm for random weight initialization [12] has been used for HAR using off-the-shelf smart-watches as well as location was added using movement information. For feature extraction and classification, histogram of gradient, and Fourier Descriptor based on Centroid signature, were adopted [17] respectively with UCI-HAR dataset. But in these contexts, researchers mentioned that the continuity of an activity, and unintended movements of the lower body part can introduce noise in the dataset which might lead to misclassification. A Dual-Path CNN-RNN that was integrated with Adam optimizer, for the purpose of achieving multi-size receptive field for better feature extraction the CNN was used. Next Recurrent Neural Network(RNN) and fully connected layers has been used to learn map between given feature and to produce output for HAR [16]. Implementation of classifiers such as MLP, SVM, and RBF which gave 81.52%, 72.18%, and 90.23% respectively [1]. Various decision trees like Random Forest, Random Tree, Hoeffding Tree, Decision Stump, J48, and REP Tree were used as the parameters for the MetaAdaboostM1 classifier [19]. The use of various sensors could capture contextual information which monitored and reflected one side of a situation using a knowledge-based approach. Thus both coarse-grained and fine-grained activity recognition were enabled [14]. However, the problem is that use of body-worn sensors [1, 19-20] can be exasperating to the elderly, and also environmental [13] sensors can create problems if more than one person is in the same space.

Khan *et al.* [34] collected 12 activity data from 20 participants and implemented 1D CNN for feature extraction, which uses spatial and discriminative features of the data. Then features are forwarded to LSTM having softmax activation function, which exhibited an accuracy of 90%. Yang *et al.* [35] used CSAS datasets, which segregated the data according to the number of members in the family to implement a sensor data contribution significance analysis. They used a Wide time-domain Convolutional Neural Network (WCNN) on all segregated groups, which gave 97% accuracy for Kyoto8.

A summary of all the major sensor-based algorithms used by several researchers is presented in Table 1.

Table 1. Summary of major sensor based algorithms used

Author	Year	Optimizer and Feature Extractor	Classifier	Accuracy in %
Sakorn ^[28]	2021	Bayesian Optimizer	CNN-LSTM	99.4
Xia ^[27]	2020	Adam Optimizer	CNN	95.8
Ahmed ^[33]	2020	SFFS	SVM	96.8
Burns ^[3]	2020	--	PTN	91.3
Taylor ^[26]	2020	--	Random Forest	96.8
Kwon ^[12]	2018	Xavier's Algorithm(Weight Initialization)	MLP	95.0
Jain ^[17]	2017	Histogram of gradient	Fourier Descriptor based on Centroid Signature	97.1
Yang ^[16]	2019	Adam optimizer and CNN feature Extractor	Dual path CNN-RRNN(DPCRNN)	99.9
Chernbumroong ^[11]	2012	--	RBF	90.2
Walse ^[19]	2016	J48(Parameter)	MetaAdaboostM1	97.8
Khan ^[34]	2022	1D CNN	LSTM	90
Yang ^[35]	2022	--	HAR_WCNN	95

Vision-based methods were also used by researchers to detect the target's posture, then the movement and speed value of the target has been determined and finally, the target's interaction with objects in the environment concluded the activity recognition [15]. A sequential Deep Trajectory Descriptor (sDTD) with CNN-RNN and CNN for feature extraction with MLP classifier has been used for vision-based action recognition [23, 24]. Researchers found three problems in vision-based data namely background changes with the self-occlusion, changing size or orientation of the subject, and different activities with similar postures. In this concern, a person-tracking system was used to control self-occlusion, multi-features to solve change in size and orientation of the subject, and embedded Hidden Markov Models (HMMs) to eliminate unnecessary data usage during testing time improve computational processing, and also increase recognition performance [25].

3. Methodology

Providing assisted living requires recognizing the activities of daily living to be most accurate. In this concern, we propose the following Genetic algorithm method for efficient selection of features and then classify them. GA is considered as a speculative global search heuristic that is inspired by the analogy of natural theory of evolution. It works faster in comparison to other methods as instead of a single point, it has the capacity to search a population of points in parallel. The diagrammatic representation of the steps involved in activity recognition is shown in Figure 1.

Our proposed method can be broadly divided into the following three steps:

- Data Acquisition
- Feature Selection
- Classifier

3.1. Data Acquisition

After conducting a study on the datasets available in public repository for the purpose of HAR, we found *Human Activities and Postural Transition (HAPT)* to be more suitable for our work as it includes a variety of activities along with different Postural transitions. HAPT is a publicly available dataset for human activity recognition that includes postural transitions. Postural transitions are transitory movements that describe the change of movement from one static posture to another [8]. This dataset has three static postures ('standing'(Sd), 'sitting'(Sn), 'lying'(L)), three ambulation activities ('walking'(W), 'walking downstairs'(WD), and 'walking upstairs'(WU)), and Postural Transition (PTs) that occur between the three existing static postures ('stand-to-sit'(SdtSn), 'sit-to-stand'(SntSd), 'sit-to-lie'(SntL), 'lie-to-sit'(LtSn), 'stand-to-lie'(SdtL), and 'lie-to-stand'(LtSd)'). To make a data collection more convenient to use and readily available smart phones are used with embedded sensors [7]. An example of a dataset collection from a postural change in sit to stand position is given in Table 2.

Use of embedded gyroscope as well as Tri-axial accelerometer in a proper frequency collected the data captured when the human body is in motion. Tri-axial linear acceleration and angular velocity signal at a sampling rate of 50Hz has been used for data collection process. To reduce the data volume, the collected data has been pre-processed by using a 3rd order low-pass Butter-worth filter with a cut-off frequency of 20Hz. In total there are 12 activity labels. For the usage of the ground truth for manual labeling, a synchronization of the experimental videos was made with the signals. The dataset contains 561 attributes and 10929 instances. The description of the dataset is given in Table 3.

3.2. Feature Selection

Feature selection is used to eliminate redundant and irrelevant data from a huge dataset which makes the classification less time-consuming. We used Genetic Algorithm (GA) as a filter-based feature selection for activity recognition. This algorithm was proposed by John Holland which is based on Darwin's survival of the fittest theory. The genetic algorithm follows natural selection and natural genetics. It focuses on an optimized solution from a population of possible solutions to a particular problem. The implemented GA's reproduction follows simple roulette wheel algorithm where each feature/attribute is given a slot size according to the fitness value. This method passes highly reproductive features to the mating pool where several combinations of features are made known as crossover to yield the best set of highly performing features.

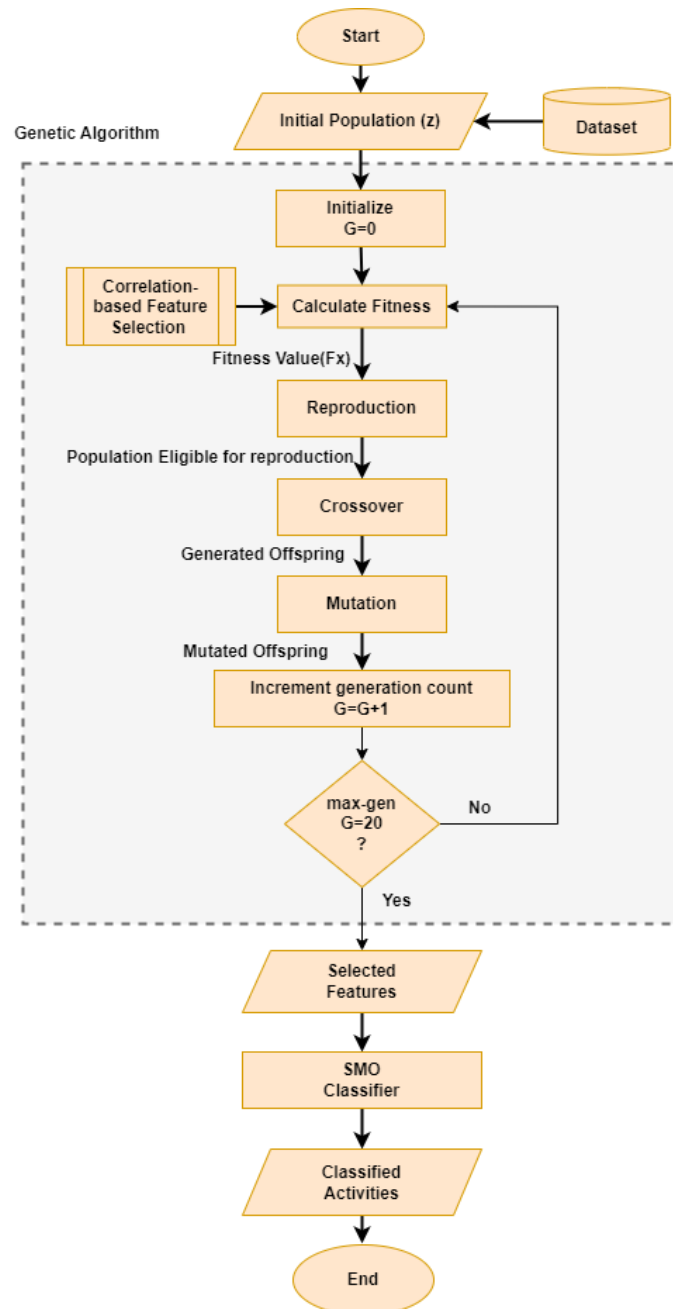


Fig. 1. Flowchart of the proposed feature section model

Table 2. Sample Dataset Collection from Sit-to-Stand posture change

Sl. No	Attributes	Value	Pictorial Instance
1	tBodyAcc-mean()-X	0.207213	
2	tBodyAcc-mean()-Y	-0.58208	
3	tBodyAcc-mean()-Z	-0.399343892	
4	tGravityAcc-correlation()-X,Y	-0.982534872	
5	tBodyAccJerk-mean()-Y	-0.068908792	
6	tBodyAccJerk-iqr()-Z	-0.9287475	
7	tBodyAccJerk-entropy()-X	0.09645113	
8	tBodyAccJerk-arCoeff()-Y,1	-0.338437577	
9	tBodyGyro-mad()-Z	-0.020437	
10	tBodyGyro-arCoeff()-Z,3	0.145486046	
11	tBodyGyroJerk-mad()-Z	-0.776968802	
12	tBodyGyroJerk-sma()	-0.85979946	
13	tBodyAccMag-max()	-0.01035274	
14	fBodyAcc-bandsEnergy()	-0.99676242	
15	fBodyAccJerk-energy()-X	-0.978527667	
16	fBodyGyro-bandsEnergy()	-0.990614018	
17	fBodyBodyGyroMag-mean()	-0.527555664	
18	angle(Z,gravityMean)	-0.27074774	
19	CLASS	SIT_TO_STAND	

Table 3. Dataset Description

Human Activities and Postural Transitions Recognition Dataset		
Instances		10929
Attributes		561
Activities	Static Postures	3
	Ambulation Activities	3
	Postural Transitions	6
	Total	12

The steps involved in GA are described as follows:

- (i) **Calculate Fitness:** Selection of pairs those will be participating in the reproduction process is selected on the basis established by the objective function. We have used Correlation-based feature selection (CFS) based fitness function for evaluation of subset and subsequent selection of optimal features. Using a heuristic evaluation function that uses correlation, ranking of attributes are done in CFS [10]. The independent attributes, those correlate with each other with the class label are first put into a separate subset using attribute vector. Now these subsets are evaluated by the function. As per the CFS method only relevant attributes show a higher degree of correlation. At the same time the probability of surfeit attributes need to be identified as being derived features they will also show a higher degree of correlation with the original attribute. The Eq. (1) is used for assessment of subset comprising 'n' features evaluating the goodness of the feature subset. Our genetic feature selection performs a search through the space of feature.

$$F_x = \frac{n\bar{C}_{fl}}{\sqrt{n + n(n-1)\bar{C}_{ff}}} \tag{1}$$

Where, F_x represents the evaluation of a subset of x that has n features

in it. $\overline{C_{fl}}$ and $\overline{C_{ff}}$ represent the average correlation value between features, class labels and between two feature respectively.

- (ii) **Reproduction:** The term reproduction is used GA for the purpose of selecting the fittest item from the population of strings hinge on the fitness value F_x calculated. In the reproduction process, based on the value F_x generated by the objective function (also called fitness function), the individual strings are copied. Here, F_x refers to the fitness value that indicates some measure of goodness, utility, or profit, that is needed to be maximized. It is the raw performance measure of the individuals provided by the fitness function based on which selection is biased. Now, these selected individuals are recombined to produce the next generation. From the participating individuals, the string values get copied according to their fitness values indicates that the string with a higher value has a higher probability of contributing one or more offspring in the next generation. Genetic information between pairs or larger groups is exchanged using the recombination operator.
- (iii) **Crossover:** The technique used to produce the offspring or the new generation string is termed as crossover. It is a process where two of the fit individuals are used for creating a fitter next generation individual. In this process any two individuals are selected from the selection pool created in the reproduction phase, and a part of their gene gets exchanged to create the new individual [21]. The crossover process uses probability index for choosing pairs from the population pool for the purposes of breeding. If a single point crossover is implemented then the value in the i^{th} position gets exchanged in between the participating pair of parents. The position i is chosen randomly anywhere within the string length, i.e., within 1 and $len-1$ where len represents the length of the string. This chosen value for i remains constant throughout the entire crossover process. For example, let's consider the following two parents represented in the form of binary strings:

$P_{a1} = 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0,$

$P_{a2} = 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0.$

And let's assume the value of $i=6$.

After applying single-point crossover the offspring created will be

$O_f1 = 1\ 0\ 1\ 1\ 1\ 1\ 1\ 0,$

$O_f2 = 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0.$

- (iv) **Mutation:** Mutation operates in the background towards ensuring the probability of appearance of a subspace within a total population to never become zero. The effect of mutation tends towards impeding the risk of convergence to a local optima in lieu of reaching the global optima. The crossover process after repeatedly applied there is a lot of chance that the resultant offspring in the successive generations would become identical to the initial pool. To avoid this, mutation process is responsible for the key functionality of making changes in the genetic pool. It makes minor changes to the offspring's genome chosen in random, thus ensuring only a small portion of the total population getting affected. Now this change applied may or may not be beneficial to the individual. Once the recombination and mutation process are applied, the newly generated mutated strings are decoded again, applied with the objective function to get a fitness value F_x that is assigned to each individual. If the applied mutation is advantageous, then the fitness gets increased and the selection mechanism approves the gene to remain in the pool but in case it

is the other way, then the mutated gene is removed from the population pool by the selection process itself. With this the performance of individuals are always expected to be on rise as only fit participants are preserved and reproduce to create an ever fitter generation removing the participants who are less fit.

The selected features are repeatedly tested and exchanged to produce best set of features. Mutation is a secondary variable with small probability alters the value of a feature's position. The mutation step might seem secondary but it helps us to prevent premature loss of important features.

Each chromosome consists of genes and the genes have a specific value and position to represent a feature. The terminology of biological genetic for AI is given in Table 4. We obtained the best feature by applying reproduction, crossover, and mutation on the initial features. This feature selection approach is more effective as the searching is applied using stochastic operators instead of deterministic rules. It searches for the solution from the population instead of a single point. These strings follow three simple operators to yield an improved result.

Table 4. Comparative terminology between Biology and AI

Biological terminology	AI terminology
Chromosome	String
Gene	Feature, Character, Or Detector
Allele	Feature Value
Locus	String Position
Genotype	Structure
Phenotype	Parameter set, Alternative solution, Decoded structure
Epistasis	Nonlinearity

3.3. Classifier

After selecting the features in this work, for the classification of 12 activities ranging from Static Postures to Postural Transitions, we used Sequential Minimal Optimization (SMO) algorithm. A divide and conquer technique is applied in SMO, where in order solve to a big Quadratic Programming (QP) problem, it is first divided into a series of smaller QP problems. These problems are further subdivided into still smaller sub-problems till the smallest possible QP is attained. Then each of these sub-problems is solved analytically. In order to avoid numerical those are time consuming, SMO approach uses an inner loop for QP optimization. The consumption of memory in SMO is linear in the training set size.

Usually, support vector classifiers are trained using SMO where the scaled polynomial kernels are applied on them. Next, standard sigmoid function is applied on the non-fitting results those are obtained from SVM for transforming them into probabilities [29]. The typical functionality of SMO involves missing value handling and replacements, conversion of attributes from nominal to binary ones and

normalization of all numeric attributes. To train a SVM, large QP requires an optimization problem.

4. Experimentation and Results

The proposed model is implemented using Waikato Environment for Knowledge Analysis (WEKA) tool of version 3.9.5 for experimentation and used python in Google collaborative for result analysis. A random partitioning of the HAPT dataset used is done in 70:30 ratios towards training and testing purpose respectively. The experimentation is based on two parts: one is classification without feature selection, and the second is classification with feature selection.

4.1. Classification without Feature Selection Model

The classifiers DL4jmlp, Decision table, Simple logistic, J48, and SMO algorithm were applied on the HAPT dataset and it was found that SMO performs best in terms of accuracy and model building time in comparison to other algorithms. The details of the test results are given in Table 5.

Table 5. Comparison between other approaches and the proposed approach

Algorithm	Accuracy with 561 features (%)	Model Building time (in Seconds)
DL4jmlp	96.27	228.3
Simple Logistic	96.91	106
Decision Table	80.60	119.9
J48	92.77	18.42
SMO	97.07	10.4

Analysis of the confusion matrix of the above algorithms showed that the classifiers made similar misclassifications except for SMO. The training time of DL4jmlp, Simple Logistic, Decision Table, J48, and SMO are 178.07s, 101.82s, 82.14s, 12.30s, and 5.62s respectively. All these measures led us to implement SMO in our final model for Activity classification. A Polynomial kernel function was used. We set SMO's complexity parameter to 1.0 and ϵ to 1.0E-12 which we found to be the optimum parameters. The validation sets were carried out using 70% training and 30% testing percentage split.

4.2. Classification with Feature Selection Model

The dataset contains 561 attributes from where we tried to reduce some irrelevant and redundant features by applying the genetic search algorithm based on its parallelism and

wide space search capabilities. In the Feature Selection Model whole volumetric data with 561 features was given as input to the feature selection algorithm i.e., Genetic algorithm. Each attribute in the dataset is considered as an individual feature which is referred to as a gene in the biological system. From the search space, a population (Z) of size 20 is chosen randomly in each batch then the fitness of the genes is calculated using the CFS. Reproduction is performed on these genes according to the fitness value after that crossover and then the mutation is performed on the chosen population. In Table 6, we have described the parameters used in the genetic search algorithm.

Our population type is numeric and a total of 20 generations (G) of off-springs are generated. We changed the value of Crossover Probability (CP) and Mutation Probability (MP) from 0.1 to 0.5 and 0.011 to 0.055 respectively to find the optimal subset of features. In the first row of Table-5, the CP and MP are 0.1 and 0.011 respectively suggests that 10% of the offspring of the next generation is created through the crossover by increasing the mutation by 1.1% in each iteration of subspace in the current generation's population. Finally, we generate 20 G of offspring, and then the final set of fittest features was selected. The details of the experimentation results are given in Table 7.

Table 6. Parameters used in Genetic Algorithm

GA Parameters	Values
Total Features	561
Population Size(Z)	20
Population Type	Numeric
Generations(G)	20
Crossover Probability(CP)	0.1-0.5
Mutation Probability(MP)	0.33-0.77

The testing of selected attributes was carried out for activity recognition using the SMO algorithm. SMO was set to Polynomial Kernel function and every selected feature set were tested with the selected classification algorithm where we can see that the feature selected using CP and MP with .3 and .033 respectively gives us the best result with minimal time.

4.3. Result Analysis and Discussion

We used performance metrics such as True Positive Rate (TPR), Precision, Recall, F1-measure, and Receiver operating Curve to analyze the efficacy of the proposed model. We also presented the performance metrics for each activity to analyze the classification and misclassification results of the proposed model.

Figure 2 shows a comparison of model building time between the original dataset and the dataset that was produced after feature selection was applied. It was observed from the experimental results that SMO and J48 take less model-building time as compared to other classifiers. DL4jMLP is an extension of MLP, which means it is capable of fitting a wide range of smooth, nonlinear functions with higher accuracy, but

due to fully connected layers, D14jMLP has a very high number of parameters, and it tends to increase the model building time hence making it cost-effective.

Table 7. Feature Selection and classification

CP	MP	Attributes	Accuracy (In %)	Model Building Time(seconds)
0.1	0.011	262	96.09	5.00
	0.022	269	96.43	5.03
	0.033	276	96.76	4.22
	0.044	285	96.18	4.12
	0.055	271	96.06	4.7
0.2	0.011	176	95.42	3.90
	0.022	278	96.58	4.16
	0.033	264	96.61	5.08
	0.044	237	96.37	3.87
	0.055	265	95.91	4.31
0.3	0.011	275	95.76	4.95
	0.022	267	96.46	4.61
	0.033	281	97.04	4.75
	0.044	287	96.24	4.69
	0.055	270	95.91	3.66
0.4	0.011	190	96.04	2.75
	0.022	212	96.34	4.91
	0.033	231	96.24	3.64
	0.044	225	96.00	4.25
	0.055	250	96.34	3.34
0.5	0.011	218	96.49	4.33
	0.022	228	95.85	3.94
	0.033	211	96.46	3.33
	0.044	259	95.73	3.25
	0.055	249	96.03	3.33

Simple Logistic performs well with linear data but our dataset contains non-linear data so the model building time is higher in comparison to J48 and SMO. The calculation of decision table is a complex multiclass classification and hence it increases the model building time. So we considered only SMO and J8 classifiers for comparison of activities in the context of postural transition. The ROC between SMO and J48 is shown in Figure 3 which indicates that SMO outperforms J48.

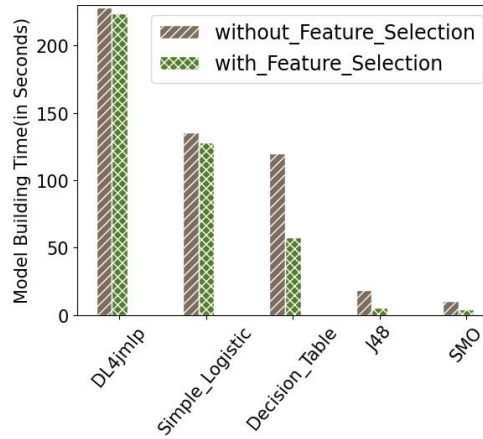


Fig. 2. Model building time analysis of the models with and without feature selection

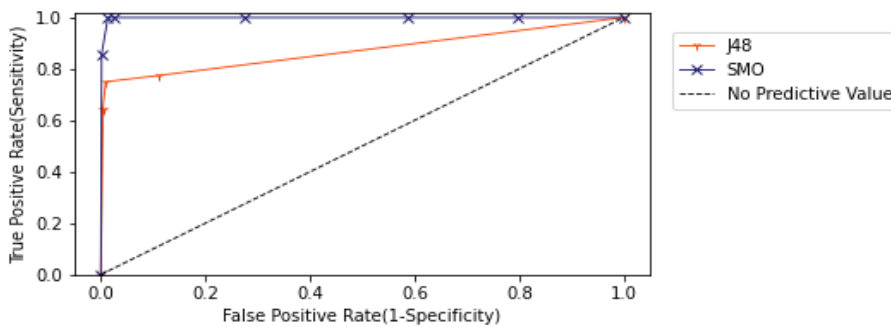


Fig. 3. ROC between J48 and SMO with 281 attributes (Sn_T_L)

The accuracy percentage of the considered classifiers was computed and is shown as in Table-6. It is found that the time complexity and accuracy performance of the SMO algorithm outperforms all other algorithms. The proposed model achieved an accuracy of 97.04% with a standard deviation of 0.22, and a mean absolute error of 0.14.

The comparison of obtained TPR, Precision, Recall, and F-measure for all the considered algorithms using selected attributes is presented in Table 8. Accuracy is the most intuitive performance measure but the ratio of correctly predicted classes to the total predicted observation i.e. precision helped to find out the percentage of our results that are relevant. From the experimental results, it was found that SMO achieved a precision of 95.7%, recall of 94.4% which indicates the ratio between correctly predicted positive observation and all observation in an actual class, and weighted average of precision and recall i.e. F-measure of 95.07%.

Table 8. Comparative results of considered classification algorithms

Algorithms	Accuracy (%)	True positive rate (%)	Precision (%)	Recall (%)	F-measure (%)
Dl4jMlp Classifier	95.21049	85.66	98.64	85.66	91.69
Simple Logistic	96.79683	94.09	96.20	94.09	95.14
Decision Table	78.76754	83.30	69.57	83.30	75.82
J48	92.31239	91.39	92.64	91.39	92.02
SMO	97.04088	94.43	95.72	94.43	95.07

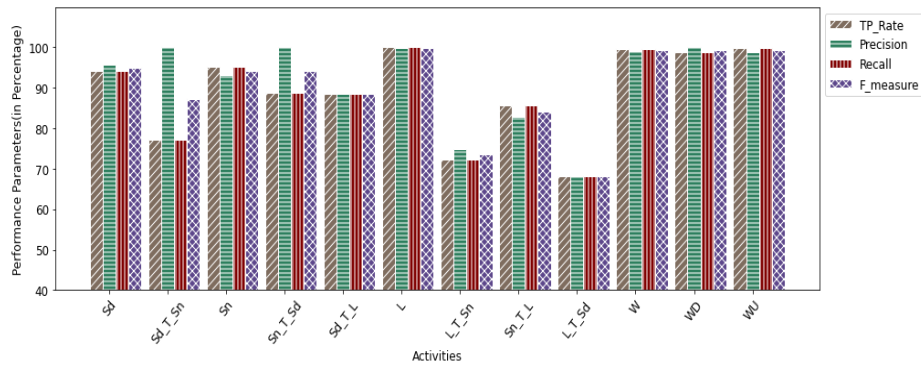


Fig. 4. Performance analysis of the SMO for all 12 activities

The efficiency of the proposed model for each of the 12 activities in terms of TP Rate, precision, recall, and F-measure is shown in Figure 4. In previous researches the higher accuracy giving algorithms, Postural Transitions were not considered. Our proposed model gives an average TP Rate of 80.01% in postural transitions. Average percentage of other performance measures such as Precision, F-measure, and Recall obtained are 85.76%, 82.66%, and 80.18% respectively.

Sd	565	0	34	0	0	0	0	0	0	0	0	0	0	0	0
Sd_T_Sn	0	17	1	0	1	0	0	1	0	0	0	0	0	0	2
Sn	25	0	509	0	0	0	0	0	0	0	0	0	0	0	0
Sn_T_Sd	0	0	1	8	0	0	0	0	0	0	0	0	0	0	0
Sd_T_L	0	0	1	0	95	0	0	4	0	0	0	0	0	0	0
L	0	0	0	0	0	595	0	0	0	0	0	0	0	0	0
L_T_Sn	0	0	0	0	0	1	21	0	7	0	0	0	0	0	0
Sn_T_L	0	0	0	0	4	0	0	24	0	0	0	0	0	0	0
L_T_Sd	0	0	0	0	0	0	0	7	15	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	477	0	0	0	0	2
WD	0	0	0	0	0	0	0	0	0	0	4	452	1	0	0
WU	0	0	0	0	0	0	0	0	0	0	0	0	1	0	460

Fig. 5. Confusion matrix of the proposed model

The obtained confusion matrix is shown in Figure 5. From Figure 4 and Figure 5, it was observed that SMO performs well even after reducing the attributes to 50%. The confusion matrix indicates that more than 80% of postural transitions are true positives with least number of attributes.

5. Proposed Cloud Architecture

Utilizing a mobile's limited resources, such as memory, energy, and embedded sensors, may lead to a decrease in the computation performance and efficiency of the system. It is also important to develop a model that is movable, consumes low battery, uses low-cost sensors, gives high accuracy, and preserves the privacy of the user. So, the use of Mobile Cloud computing is a feasible solution as it combines mobile computing, cloud computing, and wireless networks [32]. This architecture provides an environment to continuously track and monitor the activities of elderly people irrespective of the availability of an Internet connection without invading their privacy. Figure 8 shows an overview of the proposed architecture. This proposed architecture has three Layers: Data Acquisition Layer (DAL), Edge Layer (EL), Cloud Computing Layer (CCL), and Output Layer (OL). The process starts with the collection of data from various wearable gadgets continuously, offloading it to the cloud after passing it through the Edge layer for pre-processing and partial computation where the emergency situation is detected. After localizing the source of the signal, an alert is sent to the family member, caretaker, or medical practitioner with the status.

DAL: This layer deals with collecting the data that is continuously get generated from the wearable devices used by the elderly person. Functionalities for sensing real-time data and a user interface for easy status access are provided. Based on the involvement of users with the proposed system, two categories of users are identified - Active Users and Passive Users [30]. The Active User module works on the elderly person's side and activates their wearable sensors for real-time data collection. The current day's advanced wearable devices embedded with sensors have been found to be instrumental in providing personalized health services. They can be classified based on the area of usage as follows. Head-mounted devices like helmets, glasses, etc., wearable for the wrist, which includes bracelets, watches, and gloves, etc., various e-textiles such as cloths, inner wares, etc. and somatosensory modulators like various sensory control devices [22]. These devices are helpful in monitoring cardiovascular signals, salivary contents, sweat contents, physical activities, and the physiological signals of the active user. Through this monitoring, various physiological parameters such as heart rate, pulse rate, level of glucose, sodium in the blood, amount of uric acid, lactate, and potassium, as well as physical parameters such as sleep time, sitting time, daily activity, steps counts can be obtained for analysis. The collected data from the Active user module will be sent to EL directly if the Internet connection is available. In case of unavailability of the Internet, the collected data are streamed to a Local Repository, with the help of Wi-Fi or Bluetooth. Next, on availability of Internet, when the device gets network access, the data collected are uploaded to the EL and the repository is flushed. Passive User is a part of OL. This category refers to the family members/ care givers or medical practitioners who need to access the collected sensor data and are to be contacted

immediately in case of any anomalous situation identified with the activities for the active user.

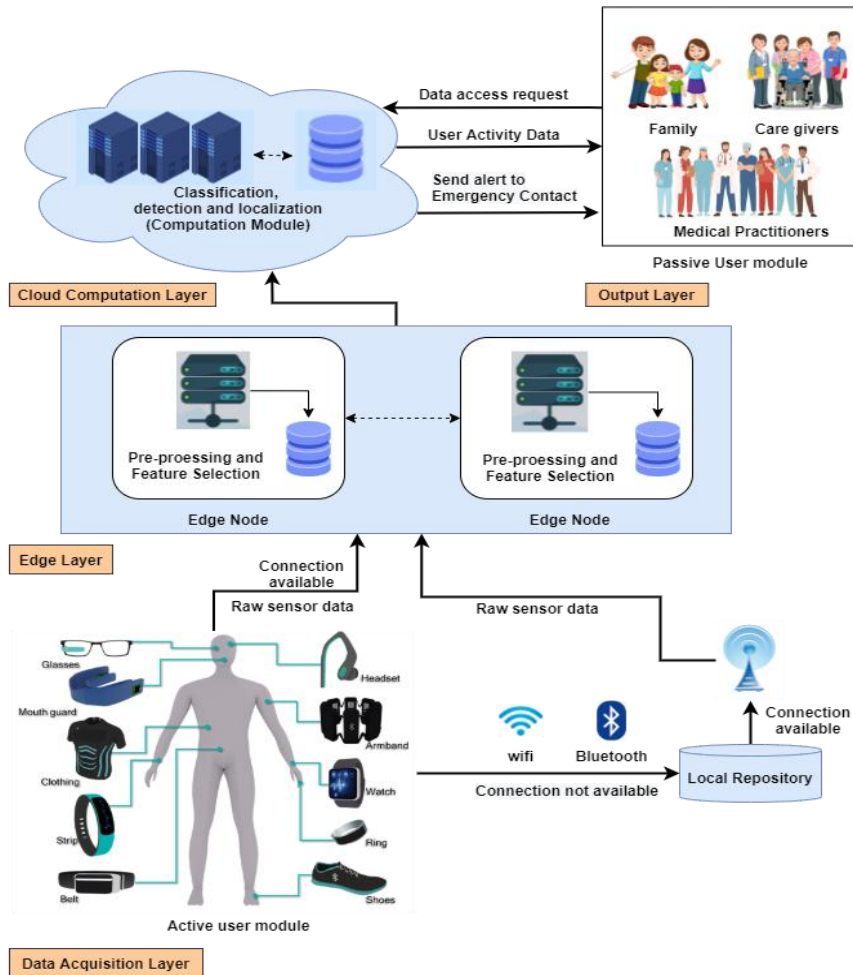


Fig. 6. Proposed HAR model in cloud architecture

EL: This layer is responsible for carrying out the following functionalities.

1) Pre-processing and cleaning: The received raw sensor data are pre-processed and data cleansing is applied for producing a usable dataset.

2) Feature Selection: From all the features available in the dataset, only relevant features are selected for further processing to ensure smoother and faster analysis of the data in the later phases.

CCL: In this layer of the proposed architecture, the data for selected features is stored, managed, and computed for activity recognition. The data is stored in the Cloud Server after the offloading from the EL. In the Computation module, the proposed classification model is deployed. When a request is sent for the status check of the elderly or patient from any passive user module, computation is performed on the stored

data, and activity status is sent to the user via the network selection module. If any unusual activity is recognized, then an alert is sent to the Passive user.

OL: This layer is responsible for all the communication between passive users and the cloud server. When the associated medical practitioners or caregivers request the user activity data for analysis, so that information regarding any improvement or deterioration in health can be derived, the data is sent to the requester in OL post-authentication process. Similarly, if any deviation in the regular pattern of received data is observed, the cloud server, after localization of the anomaly, generates an alert and sends it to the OL.

6. Conclusion and Future Work

Due to the busy lifestyle of the current generation, assisted living has become almost the need for all elderly people who spend most of their time alone at home. This paper focuses on a cost-effective, efficient model to provide elderly care without invading privacy. The proposed model is movable, consumes low battery, and uses low-cost sensors, which makes it cost-effective and affordable by general users. Genetic Algorithm is used here for feature selection, which removes many redundant features. SMO classifier is applied next on the dataset with selected features for activity classification. After the comparison of the obtained result with J48, Simple Logistic, DL4jmlp, and Decision table, it is found that the proposed model outperformed all other algorithms when applied on a dataset with feature selection using GA. For deployment of the model in the cloud, this paper also proposes a cost and privacy-preserving architecture. The highlights of the proposed architecture are sensor-based gadgets, local repositories, and edge layers. This proposed feature selection model adds to the efficiency and can be used to classify real-time data in the future. The objective of this proposed architecture is to help elderly individuals live independently within their respective homes while emergency assistance and support are provided through family members, caregivers, and associated medical practitioners.

As a future scope of work, the suggested model is proposed to be implemented in a real-life scenario. The experimentation has been carried out only on HAPT dataset. The authors propose to test the model on a custom-built real-life dataset as well.

References

1. Chernbumroong, S., Cang, S., Atkins, A., & Yu, H. Elderly activities recognition and classification for applications in assisted living. (Expert Systems with Applications, 2013), 40(5), 1662–1674., doi:10.1016/j.eswa.2012.09.004
2. World Population Ageing 2017 - Highlights ST/ESA/SER.A/397. (United Nations, Department of Economic and Social Affairs, Population Division 2017).
3. Burns, David M., and Cari M. Whyne. "Personalized Activity Recognition with Deep Triplet Embeddings.", (arXiv preprint arXiv:2001.05517, 2020)

4. Quiroz, Juan C., Amit Banerjee, Sergiu M. Dascalu, and Sian Lun Lau. "Feature selection for activity recognition from smartphone accelerometer data." (*Intelligent Automation & Soft Computing*, 2017): 1-9
5. Tang, Jiliang, Salem Alelyani, and Huan Liu. "Feature selection for classification: A review." (*Data classification: Algorithms and applications*, 2014): 37
6. Bao L., Intille S.S., Activity Recognition from User-Annotated Acceleration Data. (Ferscha A., Mattern F. (eds) *Pervasive Computing. Pervasive. Lecture Notes in Computer Science*, vol 3001. Springer, Berlin, Heidelberg, 2004.) https://doi.org/10.1007/978-3-540-24646-6_1
7. Anguita, Davide & Ghio, Alessandro & Oneto, Luca & Parra, Xavier & Reyes-Ortiz, J. A Public Domain Dataset for Human Activity Recognition using Smartphones (2013)
8. Reyes-Ortiz JL., Oneto L., Ghio A., Samá A., Anguita D., Parra X., Human Activity Recognition on Smartphones with Awareness of Basic Activities and Postural Transitions. (Wermter S. et al. (eds) *Artificial Neural Networks and Machine Learning – ICANN 2014. ICANN 2014. Lecture Notes in Computer Science*, vol 8681. Springer, Cham, 2014) https://doi.org/10.1007/978-3-319-11179-7_23
9. Liu, H., Zhou, M. and Liu, Q., An embedded feature selection method for imbalanced data classification. (*IEEE/CAA Journal of Automatica Sinica*, 2019).6(3), pp.703-715
10. Hall, M. A. & Smith, L. A., Feature subset selection: a correlation based filter approach. (*International Conference on Neural Information Processing and Intelligent Information Systems*, Berlin: Springer. 1997). pp. 855-858
11. Abolfazli, Saeid & Sanaei, Zohreh & Sanaei, Mohammad & Shojafar, Mohammad & Gani, Abdullah. Mobile cloud computing: the state-of-the-art, challenges, and future research, (*Encyclopedia of Cloud Computing*, 2015).
12. Kwon, Min-Cheol & Choi, Sunwoong. Recognition of Daily Human Activity Using an Artificial Neural Network and Smartwatch. (*Wireless Communications and Mobile Computing*, 2018). 1-9. 10.1155/2018/2618045.
13. Y. Zigel, D. Litvak and I. Gannot, A Method for Automatic Fall Detection of Elderly People Using Floor Vibrations and Sound—Proof of Concept on Human Mimicking Doll Falls, (*IEEE Transactions on Biomedical Engineering*, vol. 56, no. 12, pp. 2858-2867, 2009), doi: 10.1109/TBME.2009.2030171.
14. Chen, Liming, Chris D. Nugent, and Hui Wang., A knowledge-driven approach to activity recognition in smart homes. (*IEEE Transactions on Knowledge and Data Engineering* 24, no. 6, 2011): 961-974.
15. Brdiczka, Oliver & Langet, Matthieu & Maisonnasse, Jerome & Crowley, James. Detecting Human Behavior Models from Multimodal Observation in a Smart Home. (*Automation Science and Engineering, IEEE Transactions on*. 6. 588 – 597, 2009). 10.1109/TASE.2008.2004965. .
16. Yang, Chao, Wenxiang Jiang, and ZhongwenGuo. Time Series Data Classification Based on Dual Path CNN-RNN Cascade Network. (*IEEE Access* 7, 2019): 155304-155312.
17. Jain Ankita, and VivekKanhgad. Human activity classification in smartphones using accelerometer and gyroscope sensors. (*IEEE Sensors Journal* 18, no. 3, 2017): 1169-1177.
18. Ozcan, T., Basturk, A. Human action recognition with deep learning and structural optimization using a hybrid heuristic algorithm. (*Cluster Comput* 23, 2847–2860, 2020). doi:10.1007/s10586-020-03050-0
19. Walse, Kishor H., Rajiv V. Dharaskar, and Vilas M. Thakare. A study of human activity recognition using AdaBoost classifiers on WISDM dataset. (*The Institute of Integrative Omics and Applied Biotechnology Journal* 7, no. 2, 2016): 68-76.

20. Kutlay, Muhammed Ali, and SadinaGagula-Palalic., Application of machine learning in healthcare: Analysis on mhealth dataset. (Southeast Europe Journal of Soft Computing 4, no. 2, 2016).
21. Daniel Câmara, Evolution and Evolutionary Algorithms,Editor(s): Daniel Câmara, Bio-inspired Networking, (Elsevier, 2015), Pages 1-30, ISBN 9781785480218, doi:10.1016/B978-1-78548-021-8.50001-6.
22. Guk, & Han, Sang & Lim, Hyeongjun & Jeong, Ji hoon & Kang, Jang-Won & Jung, Sang-Chul., Evolution of Wearable Devices with Real-Time Disease Monitoring for Personalized Healthcare. (Nanomaterials, 2019). 9. 813. 10.3390/nano9060813.
23. Mo, Lingfei, Fan Li, Yanjia Zhu, and Anjie Huang., Human physical activity recognition based on computer vision with deep learning model., (IEEE International Instrumentation and Measurement Technology Conference Proceedings, 2016), pp. 1-6.
24. Shi, Yemin, YonghongTian, Yaowei Wang, and Tiejun Huang., Sequential deep trajectory descriptor for action recognition with three-stream CNN, (IEEE Transactions on Multimedia 19, no. 7, 2017): 1510-1520.
25. Jalal, Ahmad, Shaharyar Kamal, and Daijin Kim., A Depth Video-based Human Detection and Activity Recognition using Multi-features and Embedded Hidden Markov Models for Health Care Monitoring Systems, (International Journal of Interactive Multimedia & Artificial Intelligence 4, no. 4, 2017).
26. Taylor, William & Shah, Syed & Dashtipour, Kia & Zahid, Adnan & Abbasi, Qammer & Imran, Muhammad., An Intelligent Non-Invasive Real-Time Human Activity Recognition System for Next-Generation Healthcare, (Sensors. 2020) ; 20(9):2653. doi:10.3390/s20092653 2020).
27. K. Xia, J. Huang and H. Wang, LSTM-CNN Architecture for Human Activity Recognition, (IEEE Access, vol. 8, pp. 56855-56866, 2020), doi: 10.1109/ACCESS.2020.2982225.
28. Mekruksavanich, Sakorn & Jitpattanakul, Anuchit., LSTM Networks Using Smartphone Data for Sensor-Based Human Activity Recognition in Smart Homes. (Sensors. 21. 1636., 2021). 10.3390/s21051636.
29. [online] available at: <http://www.cs.tufts.edu/~ablumer/weka/doc/weka.classifiers.SMO.html>, last accessed: 21-07-2021
30. Montague, Enid, and Jie Xu, Understanding active and passive users: the effects of an active user using normal, hard and unreliable technologies on user assessment of trust in technology and co-user., (Applied ergonomics vol. 43,4 2012): 702-12. doi:10.1016/j.apergo.2011.11.002
31. Biswal, Ankita, Sarmistha Nanda, Chhabi Rani Panigrahi, Sanjeev K. Cowlessur, and Bibudhendu Pati., Human Activity Recognition Using Machine Learning: A Review., (Progress in Advanced Computing and Intelligent Engineering, 2021): 323-333.
32. Nanda, Sarmistha, Chhabi Rani Panigrahi, and BibudhenduPati., Emergency management systems using mobile cloud computing: A survey., (International Journal of Communication Systems 2020): e4619
33. Ahmed N, Rafiq JI, Islam MR. Enhanced Human Activity Recognition Based on Smartphone Sensor Data Using Hybrid Feature Selection Model (Sensors. 2020); 20(1):317. 10.3390/s20010317
34. Khan, Imran Ullah, Sitara Afzal, and Jong Weon Lee. "Human activity recognition via hybrid deep learning based model." Sensors 22, no. 1 (2022): 323.
35. Li, Yang & Guanci, Yang & Su, Zhidong & Li, Shaobo & Wang, Yang. Human activity recognition based on multienvironment sensor data. Information Fusion. 91. 47-63, 2022. 10.1016/j.inffus.2022.10.015.

36. Liu, G., Ma, J., Hu, T., & Gao, X. A feature selection method with feature ranking using genetic programming. *Connection Science*, 34(1), 1146-1168, 2022.
37. Yong, B., Wei, W., Li, K. C., Shen, J., Zhou, Q., Wozniak, M., Polap, D. & Damaševičius, R. Ensemble machine learning approaches for webshell detection in Internet of things environments. *Transactions on Emerging Telecommunications Technologies*, 33(6), e4085, 2022.
38. Hsieh, M. Y., Huang, T. C., Hung, J. C., & Li, K. C. Analysis of gesture combos for social activity on smartphone. In *Future Information Technology-II* (pp. 265-272), 2015, Springer Netherlands.
39. Hsieh, M. Y., Deng, D. J., Lin, W. D., Yeh, C. H., & Li, K. C. Self-decision activity in hierarchical wireless sensor networks. *International Information Institute (Tokyo). Information*, 15(2), 597, 2012.
40. Lin, Y., Liu, T., Chen, F., Li, K. C., & Xie, Y. An energy-efficient task migration scheme based on genetic algorithms for mobile applications in CloneCloud. *The Journal of Supercomputing*, 77, 5220-5236, 2021.

Ankita Biswal has pursued her M.Tech in Computer Science from College of Engineering and Technology, Odisha, India. Currently she is working as Assistant Professor at Centurion University, Bhubaneswar, India. She is an aspiring and enthusiastic young researcher with a keen interest in the intersection of bio-medical image processing and machine learning.

Chhabi Rani Panigrahi is currently working as an Assistant Professor in the Department of Computer Science at Rama Devi Women's University, Bhubaneswar, India. She received her Ph.D. degree in Computer Science & Engineering from IIT Kharagpur, India. She has more than 23 years of teaching and research experience. She has published several scholarly articles in journals and conferences of international repute. Her research interests include cloud computing, machine learning, and sentiment analysis.

Anukampa Behera received her MTech in Computer Science from the Biju Patnaik University of Technology, Odisha, India. She is working as an Assistant Professor at ITER, S'O'A Deemed to be University, Bhubaneswar, India and she holds more than 20 years of teaching experience. Her research focuses on finding solution to issues faced by DevOps, AI, and machine learning implementations.

Sarmistha Nanda is a Ph.D. research scholar in the Department of Computer Science at Rama Devi Women's University, Bhubaneswar, India. She worked as a JRF in NISER, Bhubaneswar, India and Research Associate at CRRI, Cuttack, India. She also served as a Senior software developer in a software firm. Her research interests include machine learning, cloud computing, and algorithms.

Tien-Hsiung Weng is currently working as a Professor in the Department of Computer Science and Information Engineering, Providence University, Taichung City, Taiwan. He received his Ph.D. in Computer Science from the University of Houston, Texas. His

research interests include parallel computing, high performance computing, scientific computing, and machine learning.

Bibudhendu Pati is an Associate Professor in the Department of Computer Science at Rama Devi Women's University, Bhubaneswar, India. He received his Ph.D. degree from IIT Kharagpur, India. He has around 25 years of experience in teaching and research. He has got several papers published in reputed journals, conference proceedings, and edited books of International repute. He is a Life Member of ISTE, Life Member of CSI, and Senior Member of IEEE.

Chandan Malu is working as the Principal Technology Architect (iCETS - Infosys Center for Emerging Technology Solutions), Infosys. He is working on Industrial Azure & Cloud Platform Engineering. He has more than 23 years of experience in Software Development. He received his B.E in Computer Science & Engineering from Utkal University and MBA from Xavier Institute of Management.

Received: June 22, 2023; Accepted: October 15, 2023.

Comparing Semantic Graph Representations of Source Code: The Case of Automatic Feedback on Programming Assignments

José Carlos Paiva^{1,2}, José Paulo Leal^{1,2}, and Álvaro Figueira^{1,2}

¹ CRACS – INESC TEC

Porto, Portugal

² DCC – FCUP

Porto, Portugal

jose.c.paiva@inesctec.pt

zp@dcc.fc.up.pt

arfiguei@fc.up.pt

Abstract. Static source code analysis techniques are gaining relevance in automated assessment of programming assignments as they can provide less rigorous evaluation and more comprehensive and formative feedback. These techniques focus on source code aspects rather than requiring effective code execution. To this end, syntactic and semantic information encoded in textual data is typically represented internally as graphs, after parsing and other preprocessing stages. Static automated assessment techniques, therefore, draw inferences from intermediate representations to determine the correctness of a solution and derive feedback. Consequently, achieving the most effective semantic graph representation of source code for the specific task is critical, impacting both techniques’ accuracy, outcome, and execution time. This paper aims to provide a thorough comparison of the most widespread semantic graph representations for the automated assessment of programming assignments, including usage examples, facets, and costs for each of these representations. A benchmark has been conducted to assess their cost using the Abstract Syntax Tree (AST) as a baseline. The results demonstrate that the Code Property Graph (CPG) is the most feature-rich representation, but also the largest and most space-consuming (about 33% more than AST).

Keywords: semantic representation, source code, graph, source code analysis, automated assessment, programming.

1. Introduction

Reasoning over graph representations of source code has innumerable applications in a wide variety of areas, ranging from software security (e.g., for detecting bugs [48, 15] and vulnerabilities [25, 80], and de-obfuscating code [52, 4, 64]) to coding assistance and computer science education (e.g., predicting variable and method names [1, 4], inferring types [36, 47], detecting similar code [74, 34, 68, 50], inferring specifications [13], and automated program repairing [14, 24]). Automated assessment has also been driving efforts into such techniques [43], as “looking” directly at the source code can enable less rigid assessment as well as richer and more formative-driven feedback than considering the result of its execution.

Achieving an effective representation for inference is a critical step in these techniques because it is an important determinant of accuracy, performance, and cost. On the one hand, encoding too much information leads to higher complexity. On the other hand, less information can reduce the accuracy of the technique. As far as the authors are aware, there is no comparison of these representations in the literature from a practical point of view, let alone for our purpose of automatically generating feedback for programming tasks. The goal of this work is to provide a detailed comparison of the most relevant semantic graph representations of source code to find out which is the most adequate for automatic feedback generation on programming assignments. Such a comparison should not only reveal which aspects of the source code each representation captures and which is the simpler representation that answers the questions at hand, but also suggest possible new ways to explore some semantic graph representations for the purpose here intended.

This work constitutes an intermediate step to building ground to the decision of which representation(s) to use in each phase of our novel automated feedback mechanism, that given an incorrect attempt (1) selects the closest correct solution from the dataset of past solutions and (2) identifies the differences between the wrong attempt and the correct one selected. To this end, we briefly describe each representation and present some recent work that uses them. We also perform feature coverage and a practical comparison of the presented semantic graph representations. The results are shown and discussed.

The remainder of this paper is organized as follows. Section 2 presents the most relevant semantic graph representations of source code. Section 3 compares the presented representations according to several aspects, using some typical program samples found in introductory programming classes. Section 4 reviews and discusses results. Finally, Section 5 summarizes the main contributions of this study. Relevant literature work is combined with the presentation of the representations and the discussion of results.

2. Intermediate Representations of Source Code

Research in the fields of program analysis and compiler design has developed a number of semantic graph representations of source code to reason about different aspects of programs. This section presents the most relevant semantic graph representations of source code for assessing submissions to programming assignments. Firstly, it introduces the AST, i.e., the “mother” of all representations. After the AST, five flow-based representations are introduced, namely the Control Flow Graph (CFG) and its interprocedural variant, the Evaluation Order Graph (EOG), the Data Flow Graph (DFG), and the Call Graph (CG). The two following subsections present two dependency-based representations, namely the Program Dependence Graph (PDG) and the System Dependence Graph (SDG). Finally, the Code Property Graph (CPG), which combines syntax, flow, and dependence information, is described.

Fig. 1 depicts the relationships between the selected semantic graph representations. The nodes illustrate the semantic graph representations, whereas the edges denote the relationships among them. There is a relationship between two representations if one is “based” (i.e., captures information from) on the other. The edge direction indicates the source of information. Syntax-based, flow-based, dependence-based, and mixed information representations are highlighted in different grey levels.

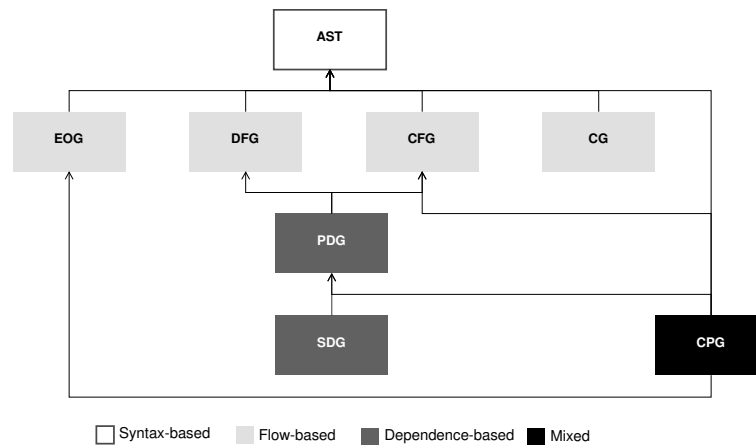


Fig. 1. Diagram view of the relationships between semantic graph representations

During the introduction of the representations, some exemplary graphs extracted from the Python programs presented in Listing 1.1, 1.2, and 1.3 are depicted. These programs are solutions to the following tasks

- T1:** enumerate even numbers from 1 to X ($X = 10$),
- T2:** calculate the factorial of Y using a recursive approach ($Y = 10$),
- T3:** sort an unordered array of Z numbers using QuickSort algorithm ($Z = 10$).

Listing 1.1. Python program to print even numbers from 1 to 10 (task T1)

```

1 X = 10
2 for i in range(1, X + 1):
3     if (i % 2) == 0:
4         print(i, end=" ")
5 print()

```

2.1. Abstract Syntax Tree (AST)

An AST is a finite, labeled, directed tree that models the syntactical structure of a program. Its internal nodes correspond to operators, whereas leaves are the operands (symbols) of the parent operator (e.g., variables, arguments, and constants). The AST preserves information regarding types and locations of variable declarations, order and components of statements, terms, operators, and their positioning in expressions, and the names and assigned values of identifiers.

Differently from parse trees, the AST omits nodes and edges from syntax rules that have no effect on the semantics of the program. For instance, grouping parentheses are

Listing 1.2. Python program to calculate the factorial of 10 using a recursive approach (task S2)

```

1 def factorial(Y):
2     if Y == 0:
3         return 1
4     return Y * factorial(Y - 1)
5
6 Y = 10
7 factorial(Y)

```

Listing 1.3. Python program to sort an unordered array of 10 numbers using Quick-sort algorithm (task S3)

```

1 def partition(arr, l, h):
2     pivot = arr[h]
3     i = l - 1
4     for j in range(l, h):
5         if arr[j] <= pivot:
6             i = i + 1
7             (arr[i], arr[j]) = (arr[j], arr[i])
8     (arr[i + 1], arr[h]) = (arr[h], arr[i + 1])
9     return i + 1
10
11 def quickSort(arr, l, h):
12     if l < h:
13         pi = partition(arr, l, h)
14         quickSort(arr, l, pi - 1)
15         quickSort(arr, pi + 1, h)
16
17 data = [8, 7, 6, 1, 0, 9, 2, 11, 10, 3]
18 Z = len(data)
19 quickSort(data, 0, Z - 1)

```


not part of the AST as their grouping is already implicit in the structure of the AST. Therefore, one way to build the AST is by performing a post-pass over the parse tree, removing nodes and edges which do not belong to the abstract syntax. An alternative way for programming languages described by context free grammars is to create the AST in the parser. In this regard, rules that contribute to the AST insert a node in the AST, whereas the symbols of the rule form edges. Figure 2 presents an AST extracted from the Python program in Listing 1.2 (note that no particular order, other than the authors' opinion, has been adopted while using the code listings in examples).

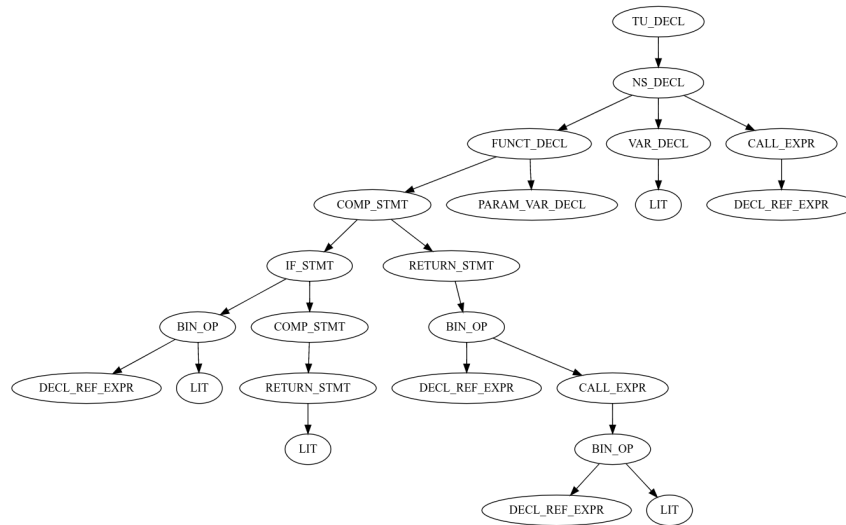


Fig. 2. Abstract syntax tree of the Python program in Listing 1.2

As ASTs are capable of maintaining a great amount of syntactic information from the original source-code, they are ideal for tasks requiring regeneration of source code, such as programming language translation [77], automatic parallelization tools [6], and program rewriting [67].

2.2. Control Flow Graph (CFG)

The Control Flow Graph (CFG) [3] models the flow of control between the basic blocks (i.e., maximal-length sequences of branch-free instructions) of a program. It consists of a directed graph, whose nodes correspond to basic blocks and edges represent control dependencies. The graph has two additional nodes, namely the entry block *ENTRY* – which is the node through which the control enters the graph – and the exit block *EXIT* – which is the node through which the control exits the graph –, such that each node has at most two successors. For instance, Figure 3 depicts the CFG extracted from the Python program in Listing 1.1.

Furthermore, two important concepts for a number of compiler optimizations and detection techniques are related to the CFG, namely post-domination and control de-

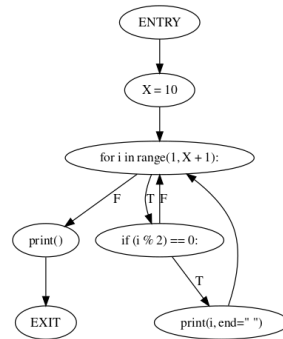


Fig. 3. Control flow graph of the Python program in Listing 1.1

pendency. The former is defined in Statement 1. For instance, in the CFG of Figure 3, node `for i in range(1, X + 1)` and node `X = 10` are post-dominated by node `print()`.

Statement 1 Consider the CFG G of a program. Let v_1 and v_2 be two nodes of G . Then, v_1 is post-dominated by a node v_2 in G , if and only if every directed path from v_1 to $EXIT$ contains v_2 .

The latter is defined in terms of post-domination relationship as in Statement 2. For instance, in the CFG of Figure 3, node `print(i, end='')` is control-dependent on node `if (i % 2) == 0`.

Statement 2 Consider the CFG G of a program and the nodes $v_3, v_4 \in G$. Then, v_4 is control-dependent on v_3 if and only if

1. there is a directed path P from v_3 to v_4 with any v_5 in P (excluding v_3 and v_4) post-dominated by v_4 and
2. v_3 is not post-dominated by v_4 .

Therefore, the CFG is an essential structure for many optimization techniques and static analysis tools, particularly because it facilitates locating inaccessible code and syntactic structures of a program such as loops. There are many recent examples of such techniques/tools in various areas of Computer Science, ranging from bug localization and vulnerability detection [12, 18, 29] to code optimization [64].

2.3. Interprocedural Control Flow Graph (ICFG)

A CFG can handle only a single procedure. For investigating the control-flow in complete programs, the Interprocedural Control Flow Graph (ICFG) is the representation to use. The ICFG connects the multiple CFGs of a program into a single graph as follows.

- For every call instruction to a function F , insert a new edge from the call to the first instruction of F , and

- from every return statement of F , insert a new edge from the return to the instruction following the call to F .

Rather than gathering information from each predecessor as usual during the analysis, the first statement of a procedure loses all data-flow information about local variables from the caller and adds details for parameters in the callee, initializing their values according to the arguments provided in the call. For instructions immediately after a call, they inherit the data-flow information from the previous instruction regarding local variables. Additionally, global variables are taken from the return instruction of the called function, whereas the variable to which the result of the function call was assigned comes from the data-flow about the returned value.

The ICFG is particularly useful for interprocedural optimizations, such as constant folding, caller-site inlining, or elimination of dead code [53, 9, 40].

2.4. Evaluation Order Graph (EOG)

An Evaluation Order Graph (EOG) – primarily introduced in Fraunhofer’s implementation of the Code Property Graph (CPG) [21] – is an intraprocedural directed graph whose vertices correspond to nodes from the AST and edges connect them in the order they would be executed when running the program, similar to the CFG. The EOG is a variant of the CFG with some subtle differences: (1) a procedure header is a node in the EOG, while the CFG only considers the first executable statement of the procedure as a node; (2) the beginning of a compound statement is a separate node in the EOG, while the CFG does not consider it; (3) the EOG represents the “if” keyword and the condition as separate nodes, while the CFG uses a single node; and (4) the EOG always ends in one (virtual) or multiple return statements (i.e., a virtual return statement is created, if the actual source code does not have an explicit one).

Therefore, comparing to the CFG in semantic terms, the EOG models the evaluation order within expressions and has operator-level precision, which enables finer granularity (more precision) in analysis. Figure 4 presents the EOG extracted from the Python program in Listing 1.2. The interprocedural version of the EOG also follows a similar approach to that of the ICFG (see Subsection 2.3), but for every call instruction to a function F , insert a new edge from the call to the F function declaration rather than the first instruction of F .

2.5. Data Flow Graph (DFG)

A Data Flow Graph (DFG) is a directed labeled graph that models the program without conditionals, i.e., using basic blocks. Edges of the graph represent paths over which data is passed, whereas nodes are the producers and consumers of such data. Dynamically, a node of the DFG accepts multiple data items from its inputs, performs some computation with them, and passes the resulting items to its outputs. For instance, Figure 5 depicts the DFG extracted from the Python program in Listing 1.1.

The DFG allows several analysis such as dead code detection, liveness analysis, and variable misuse identification [2, 26] as well as performance optimizations through parallelization [33].

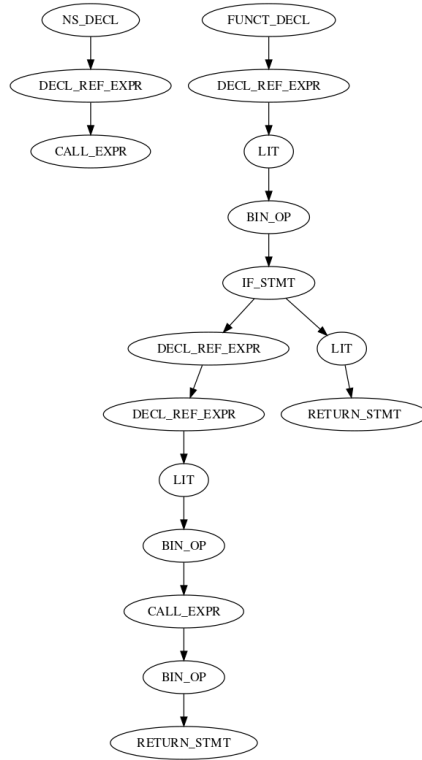


Fig. 4. Evaluation order graph of the Python program in Listing 1.2

2.6. Call Graph (CG)

The Call Graph (CG) [57] is a directed graph that connects nodes representing procedure calls to procedure start nodes. The set of nodes consists of procedure calls and starts only, whereas the edges are established between them from the start node of each routine to the call nodes. Formally, a CG can be defined, from the set of control flow graphs corresponding to the procedures of a program, as follows.

Statement 3 Consider the call graph $G = (V, E)$ and the set of control flow graphs F (so, V_f is the set of vertices of $f \in F$ and E_f its set of edges) of a program. Then, $V = V_C \cup V_S$, where V_C is the set of call instructions and V_S is the set of procedure starts in the source code, and the edge set $E \subseteq V \times V$ is defined as

$$E = \{(c, s) : c \in V_C, s \in V_S(c)\} \cup \bigcup_{(V_f, E_f) \in F} \{(s, c) : s \in V_S, c \in V_C : \exists s \rightarrow c \in V_f\} \quad (1)$$

Assuming also that $V_S(c)$ is the procedure called by $c \in V_C$. The call nodes should have exactly one incoming edge in the CFG (i.e., from the procedure it belongs to), which can be achieved by adding an empty node for nodes not meeting this requirement. Hence,

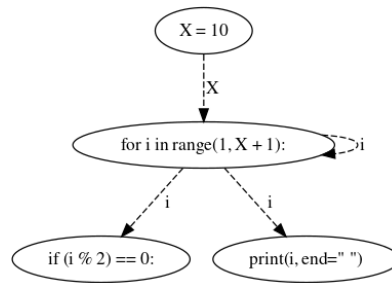


Fig. 5. Data flow graph of the Python program in Listing 1.1

each call node also has only one outgoing edge in the CFG. Concerning the CG, a call node also has exactly one incoming edge (from the start node). An example CG for the Python program in Listing 1.3 is presented in Figure 6.

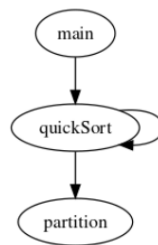


Fig. 6. Call graph of the Python program in Listing 1.3

Call graphs are widely explored in the literature as they encode symbolic and topological features of software that can be important in several different areas such as malware detection [59, 17, 23], measuring software complexity [58, 49], bug detection [69, 61], and unreachable code detection [56].

2.7. Program Dependence Graph (PDG)

A Program Dependence Graph (PDG) is a directed attributed graph that approximates program semantics, explicitly encoding control and data dependence information of the program [20]. Each node of the graph indicates a statement of the program such as an assignment, an assertion, a method call, or a return, whereas an edge represents a dependence, either regarding control or data, between two statements. For instance, given two statements s_1 and s_2 , there is a control dependence edge from s_1 to s_2 if and only if the execution of s_2 depends upon s_1 , and a data dependence edge if and only if a component assigned at s_1 is used in the execution of s_2 . Figure 7 depicts the PDG extracted from the Python program in Listing 1.1. Dashed edges represent data dependencies, whereas solid edges encode control dependencies.

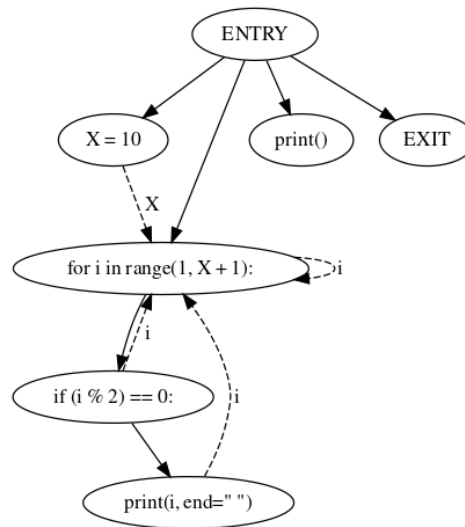


Fig. 7. Program Dependence Graph of the Python program in Listing 1.1

These graphs resort to control- and data-flow analysis to determine control and data dependencies, respectively. The former are defined in terms of the CFG (see Subsection 2.2), while the latter can be formally defined in terms of the DFG as:

Statement 4 Consider G , a DFG of a program, then nodes $s_1, s_2 \in G$ are data-dependent if and only if there is a variable v such that

- s_1 is an assignment to v ,
- s_2 uses v , and
- there is a path between s_1 and s_2 without an assignment to v .

In PDGs, dependencies connect the computationally relevant parts of the program without requiring the unnecessary control sequencing present in the CFGs (see Subsection 2.2), which means they are immune to syntactic modifications such as renaming variables, reordering statements, among others. Hence, these graphs are adequate representations to measure semantic similarity between two codes and detect semantic clones [28, 46, 38].

2.8. System Dependence Graph (SDG)

A System Dependence Graph (SDG) [27] is an extension to the PDG (see Subsection 2.7) to handle programs with multiple procedures and interprocedural calls. It models a language in which a program consists of a main procedure and a set of auxiliary procedures, to which parameters are passed by value. More formally, the SDG is a directed graph $G = (N, E)$, where N contains the nodes representing statements and predicates of the program, and E is the set of edges depicting the dependencies between them.

The SDG can be built from the PDGs of the several procedures, by connecting them at *call sites*. This consists of linking the call node c to the entry node e of the called procedure, through an edge from c to e , if there are no procedure parameters and returned values. Procedure parameters and returned values, as well as potential side effects of the called procedure, are modeled through artificial parameter nodes and edges created as follows:

- For every passed parameter p_i there exist two nodes, actual-in a_i and formal-in f_i , connected via an edge $a_i \rightarrow_{p_i} f_i$.
- For every modified parameter and returned value p_o there exist two nodes, an actual-out a_o and formal-out f_o , connected via an edge $f_o \rightarrow_{p_o} a_o$.
- Nodes f_i and f_o nodes are control dependent on the entry node e of the called procedure.
- Nodes a_i and a_o nodes are control dependent on the call node c .

In this way, it guarantees that any interprocedural effect of a procedure is propagated through the *call sites*, enabling the computation of summary information about the impact of all procedures directly or indirectly invoked. Figure 8 presents the SDG extracted from the Python program in Listing 1.2.

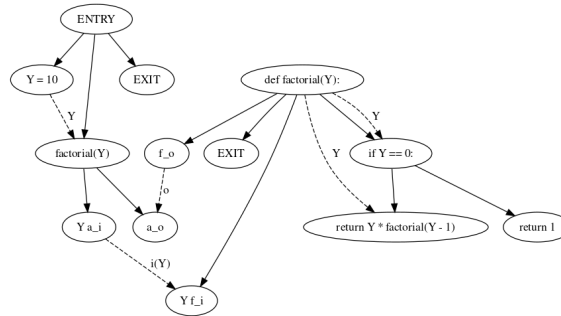


Fig. 8. System Dependence Graph of the Python program in Listing 1.2

SDGs are the basis for several applications in program analysis, ranging from program understanding [63, 45] and testing [31] to program slicing [79, 22].

2.9. Code Property Graph (CPG)

Code Property Graphs (CPGs) [76] combine three other intermediate representations, particularly ASTs, CFGs, and PDGs, into a joint data structure using the concept of property graphs – a type of graph in which edges not only connect nodes, but also carry information (e.g., a name, type, or other properties). To this end, each of the three graphs is, firstly, converted into a property graph and then combined. The AST, which provides a detailed decomposition of source code into language constructs, is expressed as a property graph $G_A = (V_A, E_A, \lambda_A, \mu_A)$, where V_A is the set of AST nodes, E_A are the corresponding

edges of the tree which are labelled as such (i.e., AST nodes) by the labelling function λ_A , and μ_A which assigns two properties, `code` and `order`, to each node. The value of the former consist of the operator or operand the node represents, whereas the latter is the index of the node in the ordered form of the tree.

Following the AST, we need to depict the CFG as a property graph $G_C = (V_C, E_C, \lambda_C, \cdot)$, where V_C relates to statements and predicates in the AST (i.e., all nodes in V_A whose property `code` corresponds to either a statement or a predicate). Moreover, the edge labeling function λ_C assigns to each edge of the property graph a label from the set $\Sigma_C = \{true, false, \epsilon\}$, according to the condition of the source node.

The next step is to transform the PDG into a property graph. As described in Subsection 2.7, the PDG represents data and control dependencies between statements and predicates and, thus, its nodes are the same as those from the CFG. In this sense, the PDG can be depicted as a property graph $G_P = (V_C, E_P, \lambda_P, \mu_P)$, where E_P is a new set of edges and $\lambda_P : E_P \rightarrow \{C, D\}$ is the corresponding edge labeling function that assigns either label C or D if it matches control and data dependencies, respectively. The former dependencies have an additional property `condition` indicating the state of the original predicate (i.e., `true` or `false`), whereas the latter dependencies have an additional property `symbol` holding the corresponding symbol.

Finally, the three property graphs are merged into a new graph $G = (V, E, \lambda, \mu)$. The set of nodes V has the same elements of V_A , whereas the set of edges E includes all edges in the three graphs. Similarly, the labeling and the property function, λ and μ respectively, are the combination of the homonymous functions of the three graphs (if they exist there). As an example, Figure 9 presents the CPG extracted from the Python program in Listing 1.2. Edges of the AST are traced in black, data dependencies are dashed, and control dependencies are in red. Property edges are omitted.

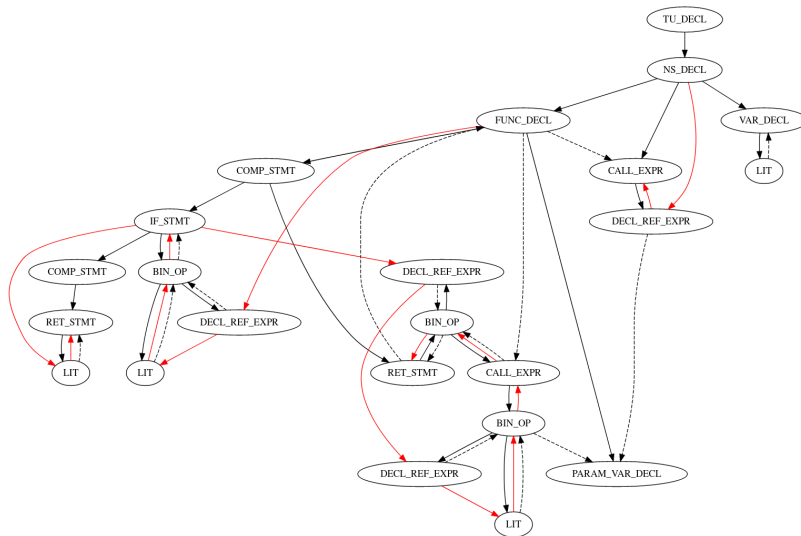


Fig. 9. Code property graph of the Python program in Listing 1.2.

The CPG was initially designed for finding vulnerabilities and detecting performance and efficiency weaknesses in source code, having already been widely and successfully applied in these tasks [76, 75, 80, 16]. The unique characteristics of this graph such as the ability to encode both syntactic and semantic value while being language-agnostic, make it attractive to other tasks (e.g., predicting names and repairing code). However, primarily due to its recency, the potential of this representation in other scenarios is yet to be explored.

3. Comparison

Determining the most effective semantic graph representation of source code for a specific task is a critical step of designing any source code analysis-dependent technique. On the one hand, the more information is encoded, the more rich and accurate can be the approach. On the other hand, too much information means higher costs, in particular, regarding execution time and memory usage. Finding the right balance between expressiveness and cost is the challenge. To this end, both the expressiveness and cost of the various semantic graph representations of source code have been compared. The following subsections present the result of each comparison.

Table 1. Comparative analysis of the expressiveness of semantic graph representations of source code.

Facet	Syntax	Flow					Dependence		Mixed
	AST	CFG	ICFG	DFG	EOG	CG	PDG	SDG	CPG
Control Dependency	Y	Y	Y	N	Y	N	Y	Y	Y
Data Dependency	Y	N	N	Y	N	N	Y	Y	Y
Multiple Procedures	Y	N	Y	Y	N	Y	N	Y	Y
Interprocedural Calls	Y	N	Y	N	N	Y	N	Y	Y
Multiple Types of Edges	N	N	N	N	N	N	Y	Y	Y
Slicing	Y	N	N	Y	N	N	Y	Y	Y
Flow-Sensitive	Y	Y	Y	Y	Y	Y	Y	N	Y
Context-Sensitive	Y	N	N	N	N	N	N	Y	Y
Inheritance & Polymorphism	Y	N	N	N	N	N	N	Y** [51]	Y
Exception Handling	Y	N	N	N	N	N	Y	Y	Y
Parameter Passing	Y	N	Y	N	N	N	N	Y	Y
Reversibility	Y*	N	N	N	N	N	N	N	Y*

* - cannot reverse to exact original source code

** - extended version

3.1. Expressiveness

From the literature and Section 2, it has been made clear that each semantic graph representation of source code described has unique capabilities. To better understand them,

Table 1 summarizes the expressiveness of the previously presented representations, including: whether it encodes control and data dependencies, if it is limited to a single procedure or captures multiple procedures, whether it can represent interprocedural calls, whether it supports multiple types of edges, whether program slicing (i.e., a reduction technique for capturing the statements that could affect the value of a variable at a specific point of interest) and flow- and context-sensitive analysis techniques can be applied, whether it can describe inheritance and polymorphism, whether it correctly deals with exception-handling constructs, whether parameter-passing information is represented, and if the source code can be recovered.

Naturally, syntax-based representations are the most feature rich, as Table 1 demonstrates for the covered facets. They represent the program with all original language constructs as well as the order between statements, typically losing only ambiguous grammar (e.g., white spaces, semicolons, trailing commas, etc) and comments. Therefore, they can even be reverted to a source code equivalent (not exactly equal) to the original. Concerning flow-based representations, these are sensitive to flow changes. They focus on either the flow of control (e.g., CFG and EOG), in which case they contain information that allows to infer control dependencies (except for the CG, which only captures procedures and interprocedural calls), or the data flow, in which case they encode details about data dependencies. Dependence-based representations, such as the PDG and the SDG, encode program dependencies, both control, and data. The SDG can represent multiple procedures, interprocedural calls, parameter passing, inheritance, and polymorphism. Both the PDG and the SDG support program slicing and exception-handling constructs. While the SDG is sensitive to the context, the PDG is flow-sensitive.

3.2. Benchmark

The extraction of semantic graph representations has been evaluated against the three sample Python programs introduced in Section 2. To this end, an existing library – Fraunhofer’s CPG library [21] –, initially designed to extract the Code Property Graph (CPG) out of source code in C/C++ (C17) and Java (Java 13), with experimental support for several other programming languages such as Python, Golang, and TypeScript, has been adapted. The changes were twofold: (1) extend the analysis with multiple passes to build the different representations that were missing (e.g., CG and CFG); and (2) develop functionality to export the representations into Comma-Separated Values (CSV), JavaScript Object Notation (JSON), and DOT [65] formats.

The JSON serialization encodes all data into a single `.json` file, containing a list of nodes, each having an ID, a list of their outgoing edges, and a key-value pair for each node property. CSV serialization stores edge and node data in separate `.csv` files. The CSV file with edges has a source and a target column, plus a column per edge property. The node descriptor file contains an ID column and an additional column for each node property being stored. DOT has been specifically designed to describe graphs and, thus, its serialization adheres to the language. For the sake of this analysis, we have used the CSV-based serialization.

Table 2 presents the results of the comparison of the semantic graph representations of source code presented in Section 2 (ICFG has been omitted as it is easily obtained from the CFG). It includes aspects such as the number of nodes, edges, and connected

components, the size of the export (in kilobytes), the build complexity (n -pass means n iterations over the AST), and whether it has edge properties.

Table 2. Results of the comparison of semantic graph representations of source code.

IR	Nodes			Edges			Components			Export Size (kB)			Build Compl.	Edge Prop.
	T1	T2	T3	T1	T2	T3	T1	T2	T3	T1	T2	T3		
AST	30	23	99	29	22	98				1.7	1.5	7.1	-	0
DFG	4	5	27	4	3	42	2	3		0.2	0.2	7.1	1-pass	1
CFG	7	8	14	8	7	14	2	3		0.3	0.3	0.3	1-pass	1
EOG	20	17	67	21	15	67	2	3		0.7	0.6	2.7	1-pass	1
CG	1	2	3	0	2	3				0.1	0.1	0.1	1-pass	0
PDG	7	9	27	10	10	72	2	3		2.1	1.6	6.6	2-pass	1
SDG	7	13	43	10	16	96				2.2	1.7	6.9	2-pass	1
CPG	30	23	99	66	52	226				2.3	2.0	9.2	3-pass	4

The most expensive representation is the CPG, as Table 2 demonstrates. In the worst case, the CPG has, in most implementations and in the one used here, the same amount of nodes as the AST, which is roughly one intermediate node for each used programming language construct plus the leaf nodes containing any parameters of these constructs. The number of edges is always less or equal to the sum of the edges in the AST, the EOG (or CFG, as in its original implementation), and the control and data dependencies between the AST nodes.

4. Discussion

There are not many works published in the literature comparing the different intermediate representations of source code. Nevertheless, a few exceptions exist. Arora et al. [7] provides a comparison of flow and dependence graphs based on features such as procedural call, slicing, sensitivity, exception handling, and test case generation. For instance, they conclude that the SDG is a feature rich representation comparing to a flow graph, supporting features like control, data and transitive dependence, single and multiple procedure, inter- and intra-procedural calls, multiple types of edges, slicing, context-sensitivity, inheritance and polymorphism, test case generation and parameter passing, whereas the flow graph is insufficient in representing data and transitive dependence, multiple procedures, inter- and intra-procedural calls, multiple types of edges, slicing, context-sensitivity, inheritance and polymorphism.

In this work, we conduct a more practical-driven comparison of various semantic graph representations of source code, measuring a number of aspects of the produced graphs. From the results presented in Section 3, it is clear that encoding more information has great impact on the graph size and, consequently, export size. On the one hand, the CPG, which combines syntactic and semantic information consumes on average 1.33 times the disk space of the AST and has about 2.3 edges for each edge of the AST, while having the same number of nodes. On the other hand, the CG is a very tiny graph whose number of nodes matches the number of subroutines of the program plus the main routine and the number of edges between any two nodes is at most 1.

The CPG is the only graph representation larger and heavier than the AST. Nevertheless, it has a single connected component, similarly to the SDG, whereas the others can have multiple connected components, typically one per procedure. Furthermore, the CPG is also the only representation (besides the AST) which can be transformed back to the original (or a similarly working) source code, as all the others lose important syntactic meaning for this transformation (e.g., EOG loses data relationships, SDG ignores instructions order, etc).

Concerning the build time (in terms of build complexity), the CPG is again the representation with the highest cost, requiring three passes over the AST: two needed to obtain the PDG (and CFG) and another to “combine” the three intermediate representations, particularly AST, CFG, and PDG. The SDG is the second in this matter, requiring a connection step after the two passes required to build the PDGs. All other representations (except the mentioned PDG) are built from the AST in a single pass.

Furthermore, we have evaluated the impact of such representations using real student submissions in multiple programming languages to programming exercises of different complexities. These submissions, taken from the PROGpedia dataset [44], refer to several years within the 2003-2020 timespan of an undergraduate Computer Science course. We have selected three correct submissions to each of four randomly chosen assignments from the dataset (6, 16, 18, and 19). The first assignment involves array manipulation. Assignment 16 requires string comparison. Assignment 18 challenges students to find the minimum cost path in a graph. Finally, assignment 19 involves depth-first search in graphs. The selected submissions for a given programming assignment adopt a similar algorithmic strategy.

Table 3 presents the export sizes of the semantic graph representations for such submissions, including the lines of code (LoC) and size of the original source code. The results support the previous conclusions regarding export sizes, i.e., the CPG has the highest cost (approximately 1.5 times more disk space than the AST), while the CG has the lowest. Regarding programming languages, the syntactic weight of Java is noticeable in most examples through a typically heavier AST. The maximum build time registered in the processed samples was 5.5 seconds for the CPG of the Java submission to programming assignment 18. However, the extraction took 4.6 seconds per submission on average.

4.1. Use Case: Dead Code Analysis

Dead code encompasses any code that has no effect on the program’s behavior, including both unused and unreachable code [55]. It is considered a harmful code smell that hinders source code comprehension [37] and downgrades its maintainability [19] and performance [66]. Examples of this anti-pattern include but are not limited to: (1) code after a return statement, in the same block; (2) variable assigned but never used; (3) methods defined but never called; and (4) code inside a conditional block whose condition is always false.

Detecting instances of the first example requires a representation that captures the possible flows of execution of the program. Therefore, the CFG (i.e., the ICFG version) is the most minimal of the presented representations that could be used. Regarding example (2), a representation that could support its detection needs to encode variables’ definition/usage lifecycle, which is exactly what DFG does. The CG, which represents the function declarations and calls between them, is the lighter representation that can

Table 3. Export sizes of semantic graph representations on PROGpedia dataset.

ID	Prog. Lang.	LoC	Size (kB)	Export Size (kB)							
				AST	DFG	CFG	EOG	CG	PDG	SDG	CPG
6	C	33	0.3	6.4	0.6	1.4	4.7	0.1	6.5	6.8	10.4
	JAVA	25	0.5	7.9	0.7	1.9	5.0	0.1	6.4	6.7	12.4
	PYTHON	20	0.4	7.0	0.6	1.3	4.6	0.1	6.1	6.3	11.2
16	C	46	0.6	7.8	0.6	1.9	5.5	0.2	5.9	5.9	11.9
	JAVA	38	0.6	8.9	0.7	2.4	5.4	0.1	5.5	5.6	13.8
	PYTHON	17	0.2	5.8	0.5	1.1	4.0	0.1	4.8	5.3	9.1
18	C	73	1.5	28.0	2.9	3.3	20.3	0.3	22.1	22.9	43.4
	JAVA	107	2.1	37.6	3.0	3.9	24.2	0.5	26.5	27.8	57.4
	PYTHON	47	1.4	27.3	1.9	3.1	17.7	0.3	20.8	21.5	37.6
19	C	98	1.9	39.3	4.4	4.8	25.7	0.2	28.2	28.4	63.9
	JAVA	87	2.1	33.0	3.0	4.6	21.4	0.5	26.1	26.7	51.9
	PYTHON	70	2.4	40.9	2.5	4.5	20.5	0.3	27.8	28.0	56.5

accurately identify instances of example (3) [42]. In this case, methods defined but never called would not be connected to the main component. Finally, instances of the fourth example demand knowledge about the flows of execution of the program and, in a few cases, tracing usage/definition of variables used in the condition (note that the condition is not evaluated, thus only obvious instances will be detected). The lighter representation fulfilling these needs is the SDG, as the PDG cannot represent multiple procedures [8]. Furthermore, the SDG is able to capture all enumerated examples of dead code.

4.2. Use Case: Measuring Program Similarity

Measuring program similarity is a common approach in automated assessment with diverse applications. On the one hand, clustering programs facilitates the generation of feedback, as the same feedback given to a previous solution in the group likely fits the new program. This is especially used in semi-automated assessment tools to reduce the number of programs that the instructors need to analyze by reusing feedback given to a prior solution to those marked as identical. Other techniques use a correct solution from the same cluster as the attempt to match against and compute the differences to generate feedback on correctness, possibly with fixes. On the other hand, plagiarism detection, which is a must-have add-on to any automated assessment system, is also solved through similarity measuring.

The similarity of programs is typically measured on intermediate representations of source code. In particular, several recent works rely on semantic graph representations due to their many advantages over textual representations, such as the ability to capture programs' strategy (e.g., iterative or recursive approach). This is the case of program clustering, where it is typically intended to group programs adopting the same strategy and/or at an identical phase of development. This is well-captured in the control flow of a program. For instance, Naudé et al. [41] clusters programs by system dependence graph similarity, with the goal of selecting the most similar model solution. This solution will be compared against the attempt to measure program correctness. Zougari et al. [81] and the prototype tool eGrader [5] follow the same approach, after a xUnit-based testing step that

ensures incorrect submissions are not marked as correct, but using a control flow graph and a combination of control and system dependence graphs as the graph representations, respectively. LAV [71] also combines such approach based on the control flow graph representation with bug finding and traditional *blackbox* testing techniques to estimate a composed grade. Nevertheless, suggesting a correction to the program code as feedback to the student, requires knowledge about the syntax [24].

In the case of plagiarism detection, these representations offer resilience against most code obfuscation techniques such as renaming variables and classes, changing order of independent instructions, among others [60, 10]. Even though plagiarism can be checked using any of these representations, resilience to a specific obfuscation method as well as the accuracy of the identified plagiarism pairs depend on the chosen representation. Therefore, several semantic representations have been successfully explored such as control flow graph [10], program dependence graph [60, 35], and system dependence graph [73].

4.3. Use Case: Automated Feedback on Correctness for Programming Assignments

Automated feedback on students' solutions to programming assignments is crucial for them to develop practical programming skills. Even though the most widely adopted assessment technique consists of executing the program with a specific input and comparing the obtained output against the expected output [43], the feedback that can be generated from such a technique is too limited for formative purposes [32]. Consequently, several static analysis approaches have been introduced that reason over semantic representations of the program's source code to derive feedback to help learners understand and improve their programs [43].

Evaluating the correctness of source code automatically through semantic representations cannot be done accurately, as any small modification (e.g., changing the order of operands in a binary comparison) can affect the result of the program. Only the AST is capable of representing such minor details, but classifying a program as correct or wrong based on its AST requires the set of all correct AST solutions (infinite) to be known. However, the logic behind correct solutions, greatly captured through semantic representations, is typically common to a group of solutions. The number of such groups can be limited, e.g., solutions to calculate the factorial belong to one of two groups: recursive and iterative. Solutions of the same group share certain characteristics such as the number of loops, number of possible execution flows, or the variable usage/definition dependencies.

Based on that, several approaches measure the similarity of a submitted program's representation to that of known solutions, i.e., the more similar the program is, the better grade it receives. In this sense, any of the representations can be used, but the CFG [70, 82] - which focuses on the structure of a program - is the most widely used. The PDG (and its extension, the SDG) are also commonly used representations for similarity-based grading [72], as besides structure information they also capture data dependencies. Even though being fully similar to a solution does not guarantee the program's correctness, if differences are found they may contain good hints on improving both the program's correctness and quality (e.g., there is a nested loop, between X and Y , which could be a simple loop).

Furthermore, after knowing the closest working solution, the ASTs of both the submitted program and the referred solution can be compared to find the minimal set of trans-

formations that convert the submitted program into the correct solution [24]. Moreover, the AST has also been used to learn program transformations that fixed students' incorrect programs in the past, aiming to suggest fixes as feedback to students with similar faults [54]. The CPG, even though compiles all information of the other representations used, has not been explored yet in generating feedback on correctness for programming assignments, due to its recency.

4.4. Summary

As these use cases demonstrate, representations may have different applications within the same task. Hence, it is essential to thoroughly define the task requirements to choose the representation that satisfies them at minimal cost. Table 4 summarizes the fulfillment of the use cases' requirements by representations. We have considered 3 levels of fulfillment: complete (●), partial (○), and none.

Table 4. Summary of the results of use case analysis

	AST	DFG	CFG	EOG	CG	PDG	SDG	CPG
Dead Code Analysis	●	○	○	○	○	○	●	●
Measuring Program Similarity	Any can be applied, depends on what one wants to compare.							
Automated Feedback on Correctness	●		○	○	○	○	○	●

5. Conclusions

Reasoning about semantic representations of source code is gaining popularity as a technique in automated assessment of programming assignments, mainly for generating feedback. Such representations are easily manageable by a computer, encoding a set of features of the source code without unnecessary "noise". As feedback provided by automated assessment tools must be real-time and meaningful, selecting or deriving an appropriate representation that optimizes both the quality of generated feedback and resource consumption is crucial. However, there is no study that summarizes what each representation offers and compares their practical costs, much less towards our goal.

This paper provides a brief summary of how most semantic graph representations are constructed from the original source code, some of their recent applications reported in the literature, and examples. In addition, a benchmark of the different representations has been conducted to evaluate their cost, in terms of size and compilation complexity, using the AST as a baseline. The results show that, on the one hand, the CPG is the most feature-rich representation, but also the largest, i.e., it requires more edges, more properties on the edges, and more space. On the other hand, the CG is the lightest representation, but it only represents the calling relationships between the subroutines of the program.

We examined three use cases to determine which representations are more appropriate for each. For the most common examples of dead code, the PDG is sufficient. As for measuring source code similarity, we can do on any representation, but to achieve a good

balance between accuracy and resilience to obfuscation, it is common to use representations that encode control flow (CFG or EOG). The last use case refers to our main target, i.e., the automatic generation of correctness feedback for programming assignments. Typically, this requires both syntactic and structural information. There are approaches that use CFG, PDG, SDG, and AST, as well as some that combine multiple representations in the same or separate phases. Therefore, we propose that our automated feedback mechanism should use (1) a control flow-based representation (e.g., CFG or EOG) to select the closest correct solution from the dataset of past solutions and (2) a representation combining syntactic and structural details to find out the meaningful differences between the wrong attempt and the correct one selected. For this, we can explore the CPG (which remains unexplored behind security topics) as the EOG is already part of it and it combines syntactic and structural data, having a cost not significantly greater than the sum of any two other representations providing such information. Furthermore, since it is a queryable graph, instructors can query it for bad coding patterns and identify common errors, among other things.

With this in mind, we have already collected and published a dataset – PROGpedia [44] – composed of the CPGs of the source code submitted on an automated assessment system to 16 programming exercises proposed in multiple years within the 2003-2020 time span to undergraduate Computer Science students. Considering the goal of our project (i.e., build a novel automated feedback mechanism), this dataset (the first of its type) is useful as it concentrates on the semantic and syntactic value of the source code of all previous attempts to solve a set of programming problems. Therefore, for any new attempt we can: (1) match it with the group of solutions following the same/similar strategy; (2) identify the closest correct solution; and (3) derive feedback from the differences that transform the current CPG into a correct one. Furthermore, these data can also help understanding the students' program development process, the relationship between the different problem-solving strategies and difficulty in achieving a correct solution, among other research problems.

Admittedly, some semantic graph representations, such as the Program-derived Semantics Graph (PSG) [30] and the Context-Aware Parse Tree (CAPT) [78], were not considered in this study due to a lack of both space and time to implement their extraction algorithm. These representations, as well as those that encode the dynamic behavior of a program, such as the Program Interaction Dependency Graph (PIDG) [11], are definitely worthy of a new study.

Several platforms for programming education could make use of these findings. These include not only automated assessment tools such as those described in the latest literature review of the area [43], but also intelligent tutoring systems [39], program visualization tools [62], among others. The improvements made to the existing CPG tool [21] are available upon request to the first author's email.

References

1. Allamanis, M., Barr, E.T., Bird, C., Sutton, C.: Suggesting accurate method and class names. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. p. 38–49. ESEC/FSE 2015, Association for Computing Machinery, New York, NY, USA (2015), <https://doi.org/10.1145/2786805.2786849>

2. Allamanis, M., Brockschmidt, M., Khademi, M.: Learning to represent programs with graphs (2018)
3. Allen, F.E.: Control flow analysis. In: Proceedings of a Symposium on Compiler Optimization. p. 1–19. Association for Computing Machinery, New York, NY, USA (1970), <https://doi.org/10.1145/800028.808479>
4. Alon, U., Zilberstein, M., Levy, O., Yahav, E.: A general path-based representation for predicting program properties. In: Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation. p. 404–419. PLDI 2018, Association for Computing Machinery, New York, NY, USA (2018), <https://doi.org/10.1145/3192366.3192412>
5. AlShamsi, F., Elnagar, A.: An automated assessment and reporting tool for introductory java programs. In: 2011 International Conference on Innovations in Information Technology. pp. 324–329. IEEE, Abu Dhabi, United Arab Emirates (April 2011)
6. Arabnejad, H., Bispo, J., Cardoso, J.M.P., Barbosa, J.G.: Source-to-source compilation targeting OpenMP-based automatic parallelization of c applications. *The Journal of Supercomputing* 76(9), 6753–6785 (Dec 2019), <https://doi.org/10.1007/s11227-019-03109-9>
7. Arora, V., Bhatia, R.K., Singh, M.P.: Evaluation of flow graph and dependence graphs for program representation. *International Journal of Computer Applications* 56, 18–23 (2012)
8. Binkley, D.: Interprocedural constant propagation using dependence graphs and a data-flow model. In: Fritzson, P.A. (ed.) *Compiler Construction*. pp. 374–388. Springer Berlin Heidelberg, Berlin, Heidelberg (1994)
9. Bouajjani, A., Esparza, J., Touili, T.: A generic approach to the static analysis of concurrent programs with procedures. In: Proceedings of the 30th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. p. 62–73. POPL '03, Association for Computing Machinery, New York, NY, USA (2003), <https://doi.org/10.1145/604131.604137>
10. Chae, D.K., Ha, J., Kim, S.W., Kang, B., Im, E.G.: Software plagiarism detection: A graph-based approach. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. p. 1577–1580. CIKM '13, Association for Computing Machinery, New York, NY, USA (2013), <https://doi.org/10.1145/2505515.2507848>
11. Cheers, H., Lin, Y.: A novel graph-based program representation for java code plagiarism detection. In: Proceedings of the 3rd International Conference on Software Engineering and Information Management. p. 115–122. ICSIM '20, Association for Computing Machinery, New York, NY, USA (2020), <https://doi.org/10.1145/3378936.3378960>
12. Chen, P., Liu, J., Chen, H.: Matryoshka: Fuzzing deeply nested branches. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. p. 499–513. CCS '19, Association for Computing Machinery, New York, NY, USA (2019), <https://doi.org/10.1145/3319535.3363225>
13. DeFreez, D., Thakur, A.V., Rubio-González, C.: Path-based function embedding and its application to error-handling specification mining. In: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. p. 423–433. ESEC/FSE 2018, Association for Computing Machinery, New York, NY, USA (2018), <https://doi.org/10.1145/3236024.3236059>
14. Devlin, J., Uesato, J., Singh, R., Kohli, P.: Semantic code repair using neuro-symbolic transformation networks (2017)
15. Dinella, E., Dai, H., Li, Z., Naik, M., Song, L., Wang, K.: Hoppity: Learning graph transformations to detect and fix bugs in programs. In: 8th International Conference on Learning Representations. OpenReview.net, Addis Ababa, Ethiopia (April 2020), <https://openreview.net/forum?id=SJEqs6EFvB>
16. Du, X., Chen, B., Li, Y., Guo, J., Zhou, Y., Liu, Y., Jiang, Y.: Leopard: Identifying vulnerable code for vulnerability assessment through program metrics. In: Proceedings of the 41st International Conference on Software Engineering. p. 60–71. ICSE '19, IEEE Press, Montreal, Quebec, Canada (2019), <https://doi.org/10.1109/ICSE.2019.00024>

17. Du, Y., Wang, J., Li, Q.: An android malware detection approach using community structures of weighted function call graphs. *IEEE Access* 5, 17478–17486 (Jun 2017)
18. Duan, Y., Li, X., Wang, J., Yin, H.: DeepBinDiff: Learning program-wide code representations for binary diffing. In: *Proceedings 2020 Network and Distributed System Security Symposium*. Internet Society, San Diego, CA, USA (2020), <https://doi.org/10.14722/ndss.2020.24311>
19. Eder, S., Junker, M., Jürgens, E., Hauptmann, B., Vaas, R., Prommer, K.H.: How much does unused code matter for maintenance? In: *2012 34th International Conference on Software Engineering (ICSE)*. pp. 1102–1111. IEEE, Zurich, Switzerland (June 2012)
20. Ferrante, J., Ottenstein, K.J., Warren, J.D.: The program dependence graph and its use in optimization. *ACM Transactions on Programming Languages and Systems* 9(3), 319–349 (Jul 1987), <https://doi.org/10.1145/24039.24041>
21. Fraunhofer AISEC: Code property graph. <https://github.com/Fraunhofer-AISEC/cpg> (2022)
22. Galindo, C., Pérez, S., Silva, J.: Slicing unconditional jumps with unnecessary control dependencies. In: Fernández, M. (ed.) *Logic-Based Program Synthesis and Transformation*. pp. 293–308. Springer International Publishing, Cham (2021)
23. Ge, X., Pan, Y., Fan, Y., Fang, C.: Amdroid: Android malware detection using function call graphs. In: *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. pp. 71–77. IEEE, Sofia, Bulgaria (July 2019)
24. Gulwani, S., Radiček, I., Zuleger, F.: Automated clustering and program repair for introductory programming assignments. In: *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation*. p. 465–480. PLDI 2018, Association for Computing Machinery, New York, NY, USA (2018), <https://doi.org/10.1145/3192366.3192387>
25. Harer, J.A., Kim, L.Y., Russell, R.L., Ozdemir, O., Kosta, L.R., Rangamani, A., Hamilton, L.H., Centeno, G.I., Key, J.R., Ellingwood, P.M., McConley, M.W., Opper, J.M., Chin, S.P., Lazovich, T.: Automated software vulnerability detection with machine learning. *CoRR abs/1803.04497* (2018), <http://arxiv.org/abs/1803.04497>
26. Hellendoorn, V.J., Sutton, C., Singh, R., Maniatis, P., Bieber, D.: Global relational models of source code. In: *8th International Conference on Learning Representations*. OpenReview.net, Addis Ababa, Ethiopia (Apr 2020), <https://openreview.net/forum?id=B1lnbRNtwr>
27. Horwitz, S., Reps, T., Binkley, D.: Interprocedural slicing using dependence graphs. In: *Proceedings of the ACM SIGPLAN 1988 Conference on Programming Language Design and Implementation*. p. 35–46. PLDI '88, Association for Computing Machinery, New York, NY, USA (1988), <https://doi.org/10.1145/53990.53994>
28. Horwitz, S.: Identifying the semantic and textual differences between two versions of a program. In: *Proceedings of the ACM SIGPLAN 1990 Conference on Programming Language Design and Implementation*. p. 234–245. PLDI '90, Association for Computing Machinery, New York, NY, USA (1990), <https://doi.org/10.1145/93542.93574>
29. Huo, X., Li, M., Zhou, Z.H.: Control flow graph embedding based on multi-instance decomposition for bug localization. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(04), 4223–4230 (Apr 2020), <https://ojs.aaai.org/index.php/AAAI/article/view/5844>
30. Iyer, R.G., Sun, Y., Wang, W., Gottschlich, J.: Software language comprehension using a program-derived semantics graph. *CoRR abs/2004.00768* (2020), <https://arxiv.org/abs/2004.00768>
31. vahab karthedath, A., Vijayan, S., S, V.K.K.: System dependence graph based test case generation for object oriented programs. In: *2020 International Conference on Power, Instrumentation, Control and Computing (PICC)*. pp. 1–6. IEEE, Thrissur, India (Dec 2020), <https://doi.org/10.1109/picc51425.2020.9362460>
32. Kyrilov, A., Noelle, D.C.: Binary instant feedback on programming exercises can reduce student engagement and promote cheating. In: *Proceedings of the 15th Koli Calling Conference on Computing Education Research*. p. 122–126. Koli Calling '15, Association for Computing Machinery, New York, NY, USA (2015), <https://doi.org/10.1145/2828959.2828968>

33. Lévai, T., Németh, F., Raghavan, B., Rétvári, G.: Batchy: Batch-scheduling data flow graphs with service-level objectives. In: Bhagwan, R., Porter, G. (eds.) 17th USENIX Symposium on Networked Systems Design and Implementation. pp. 633–649. USENIX Association, Santa Clara, CA, USA (Feb 2020), <https://www.usenix.org/conference/nsdi20/presentation/levai>
34. Li, W., Saidi, H., Sanchez, H., Schäf, M., Schweitzer, P.: Detecting similar programs via the weisfeiler-leman graph kernel. In: Kapitsaki, G.M., Santana de Almeida, E. (eds.) Software Reuse: Bridging with Social-Awareness. pp. 315–330. Springer International Publishing, Cham (2016)
35. Liu, C., Chen, C., Han, J., Yu, P.S.: Gplag: Detection of software plagiarism by program dependence graph analysis. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 872–881. KDD '06, Association for Computing Machinery, New York, NY, USA (2006), <https://doi.org/10.1145/1150402.1150522>
36. Malik, R.S., Patra, J., Pradel, M.: NI2type: Inferring javascript function types from natural language information. In: 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). pp. 304–315. IEEE, Montreal, QC, Canada (May 2019)
37. Mantyla, M., Vanhanen, J., Lassenius, C.: A taxonomy and an initial empirical study of bad smells in code. In: International Conference on Software Maintenance, 2003. ICSM 2003. Proceedings. pp. 381–384. IEEE, Amsterdam, Netherlands (Sep 2003)
38. Mehrotra, N., Agarwal, N., Gupta, P., Anand, S., Lo, D., Purandare, R.: Modeling functional similarity in source code with graph-based siamese networks (2020)
39. Mousavinasab, E., Zarifanaiey, N., Kalthori, S.R.N., Rakhshan, M., Keikha, L., Saeedi, M.G.: Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. *Interactive Learning Environments* 29(1), 142–163 (2021), <https://doi.org/10.1080/10494820.2018.1558257>
40. Müller-Olm, M., Seidl, H.: Precise interprocedural analysis through linear algebra. In: Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. p. 330–341. POPL '04, Association for Computing Machinery, New York, NY, USA (2004), <https://doi.org/10.1145/964001.964029>
41. Naudé, K.A., Greyling, J.H., Vogts, D.: Marking student programs using graph similarity. *Computers & Education* 54(2), 545 – 561 (2010), <http://www.sciencedirect.com/science/article/pii/S0360131509002450>
42. Obbink, N.G., Malavolta, I., Scoccia, G.L., Lago, P.: An extensible approach for taming the challenges of javascript dead code elimination. In: 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER). pp. 291–401. IEEE, Campobasso, Italy (March 2018)
43. Paiva, J.C., Leal, J.P., Figueira, Á.: Automated assessment in computer science education: A state-of-the-art review. *ACM Trans. Comput. Educ.* (jan 2022), <https://doi.org/10.1145/3513140>, just Accepted
44. Paiva, J.C., Leal, J.P., Figueira, Á.: Progpédia: Collection of source-code submitted to introductory programming assignments. *Data in Brief* 46, 108887 (2023), <https://www.sciencedirect.com/science/article/pii/S2352340923000057>
45. Pereira, N., Pereira, M.J.V., Henriques, P.R.: Comment-based Concept Location over System Dependency Graphs. In: Pereira, M.J.V., Leal, J.P., Simões, A. (eds.) 3rd Symposium on Languages, Applications and Technologies. OpenAccess Series in Informatics (OASICs), vol. 38, pp. 51–58. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2014), <http://drops.dagstuhl.de/opus/volltexte/2014/4558>
46. Podgurski, A., Clarke, L.: The implications of program dependencies for software testing, debugging, and maintenance. In: Proceedings of the ACM SIGSOFT '89 Third Symposium on Software Testing, Analysis, and Verification. p. 168–178. TAV3, Association for Computing Machinery, New York, NY, USA (1989), <https://doi.org/10.1145/75308.75328>

47. Pradel, M., Gousios, G., Liu, J., Chandra, S.: Typewriter: Neural type prediction with search-based validation. In: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. p. 209–220. ESEC/FSE 2020, Association for Computing Machinery, New York, NY, USA (2020), <https://doi.org/10.1145/3368089.3409715>
48. Pradel, M., Sen, K.: Deepbugs: A learning approach to name-based bug detection. *Proc. ACM Program. Lang.* 2(OOPSLA) (Oct 2018), <https://doi.org/10.1145/3276517>
49. Qingfeng, D., Kun, S., Kanglin, Y., Juan, Q.: Metrics analysis based on call graph of class methods. In: 2017 International Conference on Progress in Informatics and Computing (PIC). pp. 18–24. IEEE, Nanjing, China (Dec 2017)
50. Raghitwetsagul, C., Krinke, J.: Siamese: Scalable and incremental code clone search via multiple code representations. *Empirical Softw. Engg.* 24(4), 2236–2284 (Aug 2019), <https://doi.org/10.1007/s10664-019-09697-7>
51. Ray, M., Lal Kumawat, K., Mohapatra, D.P.: Source code prioritization using forward slicing for exposing critical elements in a program. *Journal of Computer Science and Technology* 26(2), 314–327 (Mar 2011), <https://doi.org/10.1007/s11390-011-9438-1>
52. Raychev, V., Vechev, M., Krause, A.: Predicting program properties from “big code”. In: Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages. p. 111–124. POPL ’15, Association for Computing Machinery, New York, NY, USA (2015), <https://doi.org/10.1145/2676726.2677009>
53. Reps, T., Schwoon, S., Jha, S., Melski, D.: Weighted pushdown systems and their application to interprocedural dataflow analysis. *Science of Computer Programming* 58(1), 206–263 (2005), <https://www.sciencedirect.com/science/article/pii/S0167642305000493>, special Issue on the Static Analysis Symposium 2003
54. Rolim, R., Soares, G., D’Antoni, L., Polozov, O., Gulwani, S., Gheyi, R., Suzuki, R., Hartmann, B.: Learning syntactic program transformations from examples. In: Proceedings of the 39th International Conference on Software Engineering. p. 404–415. ICSE ’17, IEEE Press, Buenos Aires, Argentina (2017), <https://doi.org/10.1109/ICSE.2017.44>
55. Romano, S.: Dead code. In: 2018 IEEE International Conference on Software Maintenance and Evolution (ICSME). pp. 737–742. IEEE, Madrid, Spain (Sep 2018)
56. Romano, S., Scanniello, G., Sartiani, C., Risi, M.: A graph-based approach to detect unreachable methods in java software. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. p. 1538–1541. SAC ’16, Association for Computing Machinery, New York, NY, USA (2016), <https://doi.org/10.1145/2851613.2851968>
57. Ryder, B.G.: Constructing the call graph of a program. *IEEE Transactions on Software Engineering* SE-5(3), 216–226 (May 1979)
58. Sawadpong, P., Allen, E.B.: Software defect prediction using exception handling call graphs: A case study. In: 2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE). pp. 55–62. IEEE, Orlando, FL, USA (Jan 2016)
59. Shang, S., Zheng, N., Xu, J., Xu, M., Zhang, H.: Detecting malware variants via function-call graph similarity. In: 2010 5th International Conference on Malicious and Unwanted Software. pp. 113–120. IEEE, Nancy, France (Oct 2010)
60. Silva, C.D.S., Ferreira da Costa, L., Rocha, L.S., Viana, G.V.R.: Knn applied to pdg for source code similarity classification. In: Intelligent Systems: 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20–23, 2020, Proceedings, Part II. p. 471–482. Springer-Verlag, Berlin, Heidelberg (2020), https://doi.org/10.1007/978-3-030-61380-8_32
61. Singh, P., Batra, S.: A novel technique for call graph reduction for bug localization. *International Journal of Computer Applications* 47(15), 1–5 (2012)
62. Sorva, J., Karavirta, V., Malmi, L.: A review of generic program visualization systems for introductory programming education. *ACM Trans. Comput. Educ.* 13(4) (nov 2013), <https://doi.org/10.1145/2490822>

63. Sridharan, M., Fink, S.J., Bodik, R.: Thin slicing. In: Proceedings of the 28th ACM SIGPLAN Conference on Programming Language Design and Implementation. p. 112–122. PLDI '07, Association for Computing Machinery, New York, NY, USA (2007), <https://doi.org/10.1145/1250734.1250748>
64. Suk, J.H., Lee, Y.B., Lee, D.H.: Score: Source code optimization & reconstruction. *IEEE Access* 8, 129478–129496 (2020)
65. The Graphviz Project: Dot language. <https://graphviz.org/doc/info/lang.html> (2022)
66. Theodoridis, T., Rigger, M., Su, Z.: Finding missed optimizations through the lens of dead code elimination. In: Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. p. 697–709. ASPLOS 2022, Association for Computing Machinery, New York, NY, USA (2022), <https://doi.org/10.1145/3503222.3507764>
67. van Tonder, R., Le Goues, C.: Lightweight multi-language syntax transformation with parser parser combinators. In: Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. p. 363–378. PLDI 2019, Association for Computing Machinery, New York, NY, USA (2019), <https://doi.org/10.1145/3314221.3314589>
68. Tufano, M., Watson, C., Bavota, G., Di Penta, M., White, M., Poshyanyk, D.: Deep learning similarities from different representations of source code. In: Proceedings of the 15th International Conference on Mining Software Repositories. p. 542–553. MSR '18, Association for Computing Machinery, New York, NY, USA (2018), <https://doi.org/10.1145/3196398.3196431>
69. Turhan, B., Kocak, G., Bener, A.: Data mining source code for locating software bugs: A case study in telecommunication industry. *Expert Systems with Applications* 36(6), 9986–9990 (2009), <https://www.sciencedirect.com/science/article/pii/S0957417408009275>
70. Vujošević-Janičić, M., Nikolić, M., Tošić, D., Kuncak, V.: Software verification and graph similarity for automated evaluation of students' assignments. *Inf. Softw. Technol.* 55(6), 1004–1016 (jun 2013), <https://doi.org/10.1016/j.infsof.2012.12.005>
71. Vujošević-Janičić, M., Nikolić, M., Tošić, D., Kuncak, V.: Software verification and graph similarity for automated evaluation of students' assignments. *Information and Software Technology* 55(6), 1004–1016 (Jun 2013), <https://linkinghub.elsevier.com/retrieve/pii/S0950584912002406>
72. Wang, T., Su, X., Wang, Y., Ma, P.: Semantic similarity-based grading of student programs. *Inf. Softw. Technol.* 49(2), 99–107 (feb 2007), <https://doi.org/10.1016/j.infsof.2006.03.001>
73. Wang, T., Wang, K., Su, X., Ma, P.: Detection of semantically similar code. *Frontiers of Computer Science* 8(6), 996–1011 (Dec 2014), <https://doi.org/10.1007/s11704-014-3430-1>
74. White, M., Tufano, M., Vendome, C., Poshyanyk, D.: Deep learning code fragments for code clone detection. In: Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. p. 87–98. ASE 2016, Association for Computing Machinery, New York, NY, USA (2016), <https://doi.org/10.1145/2970276.2970326>
75. Xiaomeng, W., Tao, Z., Runpu, W., Wei, X., Changyu, H.: Cpgva: Code property graph based vulnerability analysis by deep learning. In: 2018 10th International Conference on Advanced Infocomm Technology (ICAIT). pp. 184–188. IEEE, Stockholm, Sweden (Aug 2018)
76. Yamaguchi, F., Golde, N., Arp, D., Rieck, K.: Modeling and discovering vulnerabilities with code property graphs. In: 2014 IEEE Symposium on Security and Privacy. pp. 590–604. IEEE, Berkeley, CA, USA (May 2014)
77. Yan, Z., Qian, W., Yang, Z., Zeng, W., Yang, X., Li, A.: Tffv: Translator from eos smart contracts to formal verification language. In: Sun, X., Wang, J., Bertino, E. (eds.) *Artificial Intelligence and Security*. pp. 652–663. Springer Singapore, Singapore (2020)
78. Ye, F., Zhou, S., Venkat, A., Marcus, R., Petersen, P., Tithi, J.J., Mattson, T., Kraska, T., Dubey, P., Sarkar, V., Gottschlich, J.: Context-aware parse trees. *CoRR abs/2003.11118* (2020), <https://arxiv.org/abs/2003.11118>

79. Zhang, Y., Fu, W., Qian, X., Chen, W.: Program slicing based buffer overflow detection. *Journal of Software Engineering and Applications* 03(10), 965–971 (2010), <https://doi.org/10.4236/jsea.2010.310113>
80. Zhou, Y., Liu, S., Siow, J.K., Du, X., Liu, Y.: Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. *CoRR abs/1909.03496* (2019), <http://arxiv.org/abs/1909.03496>
81. Zougari, S., Tanana, M., Lyhyaoui, A.: Hybrid assessment method for programming assignments. In: 2016 4th IEEE International Colloquium on Information Science and Technology (CiSt). pp. 564–569. IEEE, Tangier, Morocco (Oct 2016), <http://ieeexplore.ieee.org/document/7805112/>
82. Zougari, S., Tanana, M., Lyhyaoui, A.: Towards an automatic assessment system in introductory programming courses. In: 2016 International Conference on Electrical and Information Technologies (ICEIT). pp. 496–499. IEEE, Tangiers, Morocco (May 2016)

José Carlos Paiva graduated and took his MSc in Computer Science, at the Faculty of Sciences of the University of Porto. He is currently working on his PhD in Computer Science from the same institution, entitled *Reasoning on Semantic Representations of Source Code to Support Programming Education*. He is also a research assistant at the Center for Research in Advanced Computing Systems (CRACS), a R&D unit of INESC TEC Research Laboratory, since 2014. His main research interests are automated assessment of programming tasks, gamification, e-learning and web-based learning, and machine learning.

José Paulo Leal graduated in Mathematics at the Faculty of Sciences of the University of Porto, and has a PhD in Computer Science from the same institution. His current position is auxiliary professor at the Computer Science department of the Faculty of Sciences of the University of Porto. He is also affiliated with the Center for Research in Advanced Computing Systems (CRACS), a R&D unit of INESC TEC Research Laboratory, where he is an effective member. His main research interests are technology enhanced learning, web adaptability and semantic web.

Álvaro Figueira graduated in Mathematics Applied to Computer Science, from Faculty of Sciences of the University of Porto in 1995, and took his MSc in Foundations of Advanced Information Technology, from Imperial College, London, in 1997. In 2004, he concluded his PhD in Computer Science in concurrent and distributed programming. He is currently an Assistant Professor, with tenure, at the Faculty of Sciences in University of Porto. He is also a researcher in the CRACS Research Unit where he has been leading international projects involving University of Porto, University of Texas at Austin, University of Coimbra and University of Aveiro, regarding the automatic detection of relevance in social networks. His main research interests are in the areas of text and web mining, community detection, e-learning and web-based learning and standards in education.

Received: June 15, 2023; Accepted: November 15, 2023.

MK-MSVCR: An Efficient Multiple Kernel Approach to Multi-class Classification

Zijie Dong^{1,2}, Fen Chen³, and Yu Zhang⁴

¹ School of Mathematics and Statistics,
Bigdata Modeling and Intelligent Computing research institute,
Hubei University of Education,

Second Gaoxin Road, Wuhan, 430205, China

² Hubei Key Laboratory of Applied Mathematics,
Faculty of Mathematics and Statistics,
Hubei University,
Wuhan 430062, China
zjdong07@163.com

³ School of Finance, Hubei University of Economics,
Wuhan, 430205, China
fenfen_chen@163.com

⁴ School of Mathematics and Statistics,
Hubei University of Education,
Second Gaoxin Road, Wuhan, 430205, China
romeozyu@163.com

Abstract. This paper introduces a novel multi-class support vector classification and regression (MSVCR) algorithm with multiple kernel learning (MK-MSVCR). We present a new MK-MSVCR algorithm based on two-stage learning (MK-MSVCR-TSL). The two-stage learning aims to make classification algorithms better when dealing with complex data by using the first stage of learning to generate "representative" or "important" samples. We first establish the fast learning rate of MK-MSVCR algorithm for multi-class classification with independent and identically distributed (i.i.d.) samples and uniformly ergodic Markov chain (u.e.M.c.) samples, and prove that MK-MSVCR algorithm is consistent. We show the numerical investigation on the learning performance of MK-MSVCR-TSL algorithm. The experimental studies indicate that the proposed MK-MSVCR-TSL algorithm has better learning performance in terms of prediction accuracy, sampling and training total time than other multi-class classification algorithms.

Keywords: multi-class classification, multiple kernel learning, learning rate, support vector classification and regression.

1. Introduction

Support vector machine (SVM) is an effective and famous algorithm with good generalization ability for classification. In practical problems, there are many multi-classification problems such as fault diagnosis problems, disease classification and so on. Many SVM-based methods are used to handle multi-class classification problems [2,27,18,32]. For multi-class SVM, there are two main frameworks: "all-together" method [27,18,9] and

“decomposition-reconstruction” method [3,10,15]. For the “all-together” method, we usually obtain a discrimination function by solving a single majorization problem such as AIO method [27,18,11,9]. For the “decomposition-reconstruction” method, the discrimination function is obtained by handling a series of binary classification problems, which consist of two classical approaches, “one-versus-rest” (OVR) method [3,10] and “one-versus-one” (OVO) method [15]. The disadvantage of OVR method is that almost all the binary problems are unbalanced and the shortcoming of OAO method is that for each binary category, the information of the remaining categories is neglected. Thus a new method, support vector classification and regression for multi-class classification problem, is proposed by Angulo et al. [1]. The information of all samples is used to classify by MSVCR algorithm. MSVCR has been regarded as a very important method to conquer the disadvantages of tradition multi-class classifications algorithms [1,8,7].

SVM solves nonlinear classification problems by introducing kernel functions, called kernel methods. Although the kernel method can be used to solve some complex problems, it brings many interdisciplinary challenges in statistics, optimization theory and applications [1]. Choosing the optimal kernel and its parameters often has to be decided by a human user with prior knowledge of the data. In other words, the classical classifier is based on a single kernel, while in practice, the ideal classifier is usually based on the combination of multiple kernels, i.e. multiple kernel learning.

Therefore, multiple kernel learning has been widely concerned and studied. For example, Lanckriet et al. [16] introduced the method which learns the kernel matrix with semi-definite programming to search the optimal of unrestricted kernel combination weights and showed that multiple kernel learning is comparable with the best soft margin SVM for radial basis function (RBF) kernel. Luo et al. [20] introduced a theoretically motivated and efficient online learning algorithm for the multiple kernel learning problem. In recent years, the multiple kernel learning method of iteratively updating kernel weights to obtain the optimal kernel combination has been successfully applied in many fields. For example, Chavaltada et al. [5] proposed a method of automated product categorisation by using multiple kernel learning to improve feature combination in e-commerce. Wilson et al. [28] applied multiple kernel learning to genomic data mining and prediction. Lauriola et al. [17] enhanced deep neural networks via multiple kernel learning, and the method proposed in [17] gave an effective way to design the output computation in deep networks. Wang et al. [26] propose a novel depth-width-scaling multiple kernel learning (DWS-MKL) algorithm that can adapt to data of different dimensions and sizes. In addition, machine learning algorithms are used in various fields, such as weather forecasting and smart city construction [4,33].

However, when the sample size is large, the complexity of the multi-core learning algorithm is very high. This means that although multiple kernel learning methods have good learning performance, and multiple kernel learning methods are usually very time-consuming and difficult to achieve even when the training sample size is large. Then a problem is posed:

How to reduce the algorithmic complexity of the multiple kernel learning method while maintaining the better classification accuracy?

To solve this problem, we present the idea of two-stage learning for multiple kernel algorithm in this paper. The reasons of introducing two-stage learning are as follows: First, the capacity of data is growing rapidly and the scale of data is getting larger and larger, so

the classical multiple kernel learning methods usually time-consuming and even difficult to implement in the case of a large training sample size. Second, the larger the amount of data, the lower the value density of data will be, which means that there are many noise samples in big data. A large number of machine learning experiments indicate that the noise samples will not only lead to the increase of storage space, but also affect the accuracy and efficiency of learning. By the statistical learning theory in [25], the samples that are closer (or the closest) to the interface of different classes datasets are “important” samples for classification problem. Thus, the two-stage learning aim to generate the “representative” or “important” samples by using the first stage learning. To our knowledge, this is the first algorithm of multiple kernel learning method to deal with multi-class classification non-i.i.d. samples. Therefore, in this paper we analyse the generalization of MK-MSVCR method for u.e.M.c. samples and i.i.d. observations, respectively. The main innovations of this paper can be stated as follows.

- The generalization bound of MK-MSVCR based on u.e.M.c. samples is obtained and the optimal learning rate is established.
- A new MK-MSVCR algorithm, MK-MSVCR-TSL is proposed. The numerical experiments show that the proposed algorithm has competitive performance.

The rest of this article is arranged as follows. Section 2 formulates the classical MSVCR with multiple kernel learning. Section 3 introduces a new MK-MSVCR algorithm based on two-stage learning (MK-MSVCR-TSL) and analyzes algorithmic complexity. The main theoretical results of the proposed MK-MSVCR with u.e.M.c. and i.i.d. samples are given in Section 4. The numerical experimental studies are presented in Section 5. Finally, we conclude this paper in Section 6.

2. MK-MSVCR learning machine

We assume that the training set $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^m$ are drawn from an unknown probability distribution ρ defined on the space $\mathcal{Z} = X \times Y$, where X is the input space and Y is the corresponding output space. For multi-class problems, we usually assume that $y_i \in \{1, \dots, k\}$, where k is the number of class.

The classifier of MSVCR algorithm depends on the reproducing kernel Hilbert space (RKHS) \mathcal{H}_K . Furthermore, for the given training set \mathbf{z} , a decision function $\phi(x)$ is found with outputs $\{-1, 0, +1\}$:

$$\phi(x_j) = \begin{cases} +1, & j = 1, \dots, m_1 \\ -1, & j = m_1 + 1, \dots, m_1 + m_2 \\ 0, & j = m_1 + m_2 + 1, \dots, m. \end{cases}$$

Without loss of generality, $m_{12} = m_1 + m_2$ patterns corresponds the case of two classes to be separated, and $m_3 = m - m_{12}$ patterns corresponds the case of rest classes, which will be labeled 0. The multi-class classification framework of MSVCR can be stated as

follows:

$$\begin{aligned}
f_{\mathbf{z}} &= \arg \min_{f \in \mathcal{H}_K} \lambda \|f\|_K^2 + \frac{1}{m_{12}} \sum_{j=1}^{m_{12}} \xi_j + \frac{1}{m_3} \sum_{j=m_{12}+1}^m (\varphi_j^+ + \varphi_j^-) \\
s.t. \quad & y_j \left(\sum_{i=1}^m \alpha_i K(x_j, x_i) \right) \geq 1 - \xi_j, j = 1, \dots, m_{12}, \\
& -\varepsilon - \varphi_j^- \leq \sum_{i=1}^m \alpha_i K(x_j, x_i) \leq \varepsilon + \varphi_j^+, j = m_{12} + 1, m_{12} + 2, \dots, m, \\
& \xi_j, \varphi_j^+, \varphi_j^- \geq 0, 0 \leq \varepsilon < 1.
\end{aligned}$$

In practice, an ideal classifier is usually based on a combination of multiple kernels. Thus we also present the MSVCR algorithm based on multiple kernels learning as follows. We assume that there are n positive definite kernels K_1, \dots, K_n , each RKHS \mathcal{H}_p is associated with a Mercer kernel $K_p : X \times X \rightarrow \mathbb{R}$, $1 \leq p \leq n$. By the reproducing property of \mathcal{H}_p , we have $\langle K_{p,x}, g \rangle_{K_p} = g(x)$, $\forall x \in X, \forall g \in \mathcal{H}_p$. Let $\mathcal{C}(X)$ be the space of continuous functions with the norm $\|f_p\|_\infty = \sup_{x \in X} |f_p|$ and $\kappa = \sup_{x \in X} \sqrt{K_p(x, x)}$. It follows that $\|f_p\|_\infty \leq \kappa \|f_p\|_{K_p}$, $\forall f_p \in \mathcal{H}_p, 1 \leq p \leq n$. We finally use the multiple kernel space $\bar{\mathcal{H}}_K = \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_n}$. $\bar{\mathcal{H}}_K$ is an RKHS with the kernel $\bar{K}(\cdot, x)$, which has following form:

$$\bar{K}(\cdot, x) = \sum_{p=1}^n d_p K_p(\cdot, x),$$

where $\sum_{p=1}^n d_p = 1, d_p \geq 0$. Therefore, any $f \in \bar{\mathcal{H}}_K$ has the form $f = \sum_{p=1}^n d_p f_p, f_p \in \mathcal{H}_p$. The MK-MSVCR algorithm depends on RKHS $\bar{\mathcal{H}}_K$, which is defined as

$$\begin{aligned}
f_{\mathbf{z}} &= \arg \min_{f \in \bar{\mathcal{H}}_K} \lambda \sum_{p=1}^n d_p \|f_p\|_{K_p}^2 + \frac{1}{m_{12}} \sum_{j=1}^{m_{12}} \xi_j + \frac{1}{m_3} \sum_{j=m_{12}+1}^m (\varphi_j^+ + \varphi_j^-) \quad (1) \\
s.t. \quad & y_j \left(\sum_{i=1}^m \alpha_i \bar{K}(x_j, x_i) \right) \geq 1 - \xi_j, j = 1, \dots, m_{12}, \\
& -\varepsilon - \varphi_j^- \leq \sum_{i=1}^m \alpha_i \bar{K}(x_j, x_i) \leq \varepsilon + \varphi_j^+, j = m_{12} + 1, m_{12} + 2, \dots, m, \\
& \xi_j, \varphi_j^+, \varphi_j^- \geq 0, 0 \leq \varepsilon < 1, \\
& \sum_{p=1}^n d_p = 1, d_p \geq 0.
\end{aligned}$$

Here, λ controls the complexity of the function in the multiple kernel space. For simplicity, we take $d_p = 1/n$ in this paper. The corresponding decision function $\phi_{\mathbf{z}}(x)$ of MK-MSVCR algorithm (1) is defined as

$$\phi_{\mathbf{z}}(x) = \begin{cases} +1, & \text{if } f_{\mathbf{z}}(x) \geq \varepsilon_0 \\ -1, & \text{if } f_{\mathbf{z}}(x) \leq -\varepsilon_0 \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where ε_0 is a threshold value.

Define loss function $V(y_j, f(x_j)) = C_1(1 - y_j f(x_j))_+ \cdot \mathbf{1}_{\{y_j \neq 0\}} + C_2(y_j - f(x_j))^2 \cdot \mathbf{1}_{\{y_j = 0\}}$, where C_1, C_2 are two positive constants. The MK-MSVCR algorithm (1) can be written as

$$f_{\mathbf{z}} = \arg \min_{f \in \mathcal{H}_K} \left\{ \frac{1}{m} \sum_{i=1}^m V(y_i, f(x_i)) + \lambda \|f\|_K^2 \right\}, \quad (3)$$

where $\|f\|_K^2 = \frac{1}{n} \sum_{p=1}^n \|f_p\|_{K_p}^2$ is regularization term and $\lambda > 0$ is a regularization parameter. The empirical risk and the corresponding generalization error are defined as $\mathcal{E}_{\mathbf{z}}(f) = \frac{1}{m} \sum_{i=1}^m V(y_i, f(x_i))$, $\mathcal{E}(f) = E[V(y, f(x))] = \int_{\mathcal{Z}} V(y, f(x)) d\rho$, then the MK-MSVCR algorithm (3) can be rewritten as $f_{\mathbf{z}} = \arg \min_{f \in \mathcal{H}_K} \{ \mathcal{E}_{\mathbf{z}}(f) + \lambda \|f\|_K^2 \}$.

3. Algorithm and Algorithmic complexity

In this section, we present the MK-MSVCR algorithm with two-stage learning (MK-MSVCR-TSL) and then we analyze the algorithmic complexity of the MK-MSVCR-TSL algorithm.

3.1. MK-MSVCR-TSL algorithm

Inspired by the works in [22,21,13,34], We combine the MK-MSVCR algorithm that the multiple kernel multi-class classification algorithm for processing more complex data with two-stage learning (MK-MSVCR-TSL) to improve the classification rate without reducing the classification accuracy. Now the proposed MK-MSVCR-TSL algorithm can be stated as follows.

For simplicity, all the experimental results of this paper are based on $q = 1$. In the preprocessing step, all data are normalized to avoid the influence of numerical range on characteristic attributes and make numerical calculation easier [14,19]. We use a random process to divide each data sets into two different parts, where four quarters is divided into the training set D_{train} , one quarter is divided into the test set D_{test} . Let k be the number of classed, m be the total number of training samples. For MK-MSVCR algorithm based on randomly independent samples, we sample m training samples that are drawn randomly from the given training set D_{train} and denote it as \mathbf{z} . Training the sample set \mathbf{z} by algorithm (3) and obtain a classifier $\phi_{\mathbf{z}}$. We test $\phi_{\mathbf{z}}$ on the given testing set, and calculate the corresponding misclassification rate. For MK-MSVCR-TSL algorithm, we state the algorithm as follows: (i) for the first stage, we sample randomly $N = m/2$ training samples that are drawn randomly from the given training set D_{train} and denote it as \mathbf{z}^0 . Training \mathbf{z}^0 by algorithm (3) and obtain a classifier $\phi_{\mathbf{z}^0}$. (ii) for the second stage, we use $\phi_{\mathbf{z}^0}$ to define the acceptance probabilities and generate the Markovian chain samples \mathbf{z}^1 , which consist of N samples. Training the sample set \mathbf{z}^1 by algorithm (3) and obtain a classifier $\phi_{\mathbf{z}^1}$. (iii) We test $\phi_{\mathbf{z}^1}$ on the given testing set, and calculate the corresponding misclassification rate.

Remark 1. To have a better understanding of Algorithm 1, we give the following remarks. Since we only have the training set D_{train} , to define the transition probabilities of Markovian resampling, we first draw randomly a training set \mathbf{z}^0 with N training samples, and

Algorithm 1: MK-MSVCR-TSL

Input: D_{train}, m, N .
Output: $\phi_{\mathbf{z}}(x)$.
For each sample $z \in D_{train}$, compartmentalize the training set into three parts: zero class, positive class, negative class;
(the first stage) generate randomly N samples $\mathbf{z}^0 := \{z_t\}_{t=1}^N$ from D_{train} ;
get a model $\mathbf{f}_{\mathbf{z}^0}$ by algorithm (3) with \mathbf{z}^0 and its corresponding classifier $\phi_{\mathbf{z}^0}$;
(the second stage)
generate randomly a sample z_a from D_{train} ; $\mathbf{z}^{i+1} \leftarrow \{z_a\}$;
 $count_{y_a} \leftarrow count_{y_a} + 1$;
repeat
 generate another random sample z_b from D_{train} ;
 $P = e^{-\ell(\phi_{\mathbf{z}^i}, z_b)} / e^{-\ell(\phi_{\mathbf{z}^i}, z_a)}$;
 if $P = 1$ and $y_a = y_b$ and $count_{y_b} < N/k$ **then**
 | add z_b to \mathbf{z}^{i+1} ; $count_{y_b} \leftarrow count_{y_b} + 1$;
 else if ($P = 1$ and $y_a \neq y_b$ and $count_{y_b} < N/k$) or ($P < 1$ and $count_{y_b} < N/k$) **then**
 | add z_b to \mathbf{z}^{i+1} ; $count_{y_b} \leftarrow count_{y_b} + 1$;
 end
 If ℓ candidate samples z_b can not be accepted continually, then accepting z_b ; $z_a \leftarrow z_b$;
until $size(\mathbf{z}^{i+1}) \geq N$;
get the model with \mathbf{z}^{i+1} , and obtain the decision function $\phi_{\mathbf{z}^{i+1}}$ by equation (2);
return $\phi_{\mathbf{z}} = \phi_{\mathbf{z}^q}$;

obtain a preliminary learning classifier $\phi_{\mathbf{z}^0}$. Then we use the information of $\phi_{\mathbf{z}^0}$ to define the transition probabilities P (or P_1, P_2, P_3 defined in Algorithm 1). Since these acceptance probabilities P, P_1, P_2, P_3 are positive, we can obtain an u.e.M.c sequence \mathbf{z}^1 . Specially, if the acceptance probabilities P, P_1, P_2, P_3 are equal to 1, which is the case of random resampling. This reflects that the classical MSVCR algorithm based on i.i.d. samples can be regarded as the special case of Algorithm 1 with $q = 0, N = m$, and the acceptance probabilities P, P_1, P_2, P_3 are equal to 1. This implies Algorithm 1 extended the classical MSVCR algorithm introduced in [1] from i.i.d. samples to non-i.i.d. samples. In addition, since as the value $e^{-\ell(\phi_{\mathbf{z}^i})}$ of the current sample z_a is smaller, the acceptance probabilities will be smaller, which implies that the candidate sample z_b will be accepted with a smaller probability, which means that generating the Markovian samples $\mathbf{z}^j (1 \leq j \leq q)$ will be time-consuming. To quickly draw the Markovian samples \mathbf{z}^1 , we use the technical parameters $\ell = 30$ in the following all the experimental results.

3.2. Explanations of algorithm

(i) *Comparing Algorithm 1 with algorithm introduced in [7]*

Dong et al. aim to extend the case of i.i.d. samples to non-i.i.d. samples, and study the theory and algorithm of non-i.i.d. multi-classification methods in [7]. However, we are now working to extend single kernel learning to multiple kernels learning, studying the theory and algorithm of non-i.i.d. multiple kernels multi-classification methods. SVM can solve nonlinear classification problem by kernel method. Choosing the optimal kernel from a set of candidates and its parameters is a central choice, which usually must be

made by a human user using the prior knowledge of data. In other words, the classical kernel-based classifiers are based on a single kernel. With the advent of the big data era, the data is diversified and the data characteristics are complicated. In practice, an ideal classifier is usually based on a combination of multiple kernels. For complex big data samples, we hope to use multiple kernels learning to improve classification efficiency without increasing classification time. Dong et al. pointed out in [7,9] that a proper q value can reduce the sampling and training total time of the algorithm without reducing the accuracy for Markovian resampling. If the total time is a concern, we should choose a smaller q value. Multiple kernels learning can effectively reduce the misclassification rates. However, because the combined kernel learning takes time, in order to effectively reduce the total time, we choose a smaller q value, let $q=1$ in this paper. So we propose MK-MSVCR-TSL method. The effectiveness of our method is also proved in the following experimental comparison.

(ii) *Comparing Algorithm 1 with algorithm introduced in [1]*

Comparing Algorithm 1 with algorithm introduced in [1], we can find that the differences are obvious: First, Algorithm 1 is a multiple kernels algorithm while the algorithm presented in [1] is a single kernel learning. This implies that Algorithm 1 extended the algorithm presented in [1] from a single kernel to the case of multiple kernels. In other words, the algorithm of [1] is a special case of Algorithm 1 proposed in this paper. Second, the algorithm presented in [1] is for the case that the samples are random and independent. However, our proposed Algorithm 1 is designed for not random and independent samples. This implies that Algorithm 1 improve algorithm of [1].

(iii) *Comparing Algorithm 1 with algorithm of [31]*

we can find that although Algorithm 1 has many steps similar to that of [31] and the two same technical parameters are adopted, the differences are obvious: First, Algorithm 1 is a multiple kernels algorithm while the algorithm presented in [31] is a SVM algorithm with a single kernel. Second, Algorithm 1 is a multi-class classification algorithm while [31] is a two-class classification algorithm. This implies that algorithm of [31] can be regarded as the special case of Algorithm 1 with $k = 2$ and $N = m/2$. Third, the total size of training samples for [31] is $2m$, which is double times of the size m for the classical algorithm. This implies that compared to the classical SVM, algorithm of [31] is time-consuming. While the learning process of Algorithm 1 can be seen as 2 times “batch learning”, and the total size of training samples is $m = 2N$, which is same as the classical SVM.

3.3. Algorithmic complexity

There are m samples. Here k is the number of class. In Algorithm 1, the complexity of a single kernel MSVCR is $O(\frac{K(K-1)}{2}m^3)$. In this paper, we choose the mean weights as the kernel weights of MK-MSVCR. Therefore, the complexity of MK-MSVCR algorithm is about $O(\frac{K(K-1)}{2}m^3)$. But in Algorithm 1, we divided m training samples into $q + 1$ pieces, thus, the complexity of Algorithm 1 is about $O(\frac{K(K-1)}{2}(\frac{m}{q+1})^3)$. If we assume $(q + 1) \approx m^\gamma$ for any $\gamma > 0$, it is obvious that the complexity of our proposed MK-MSVCR-TSL algorithm in this paper is lower than that of the classical MK-MSVCR.

4. Estimating the Generalization Bounds

In this section, our target is to bound the generalization of MK-MSVCR-TSL algorithm. In this paper we first consider the case of uniformly ergodic Markov chain (u.e.M.c.) observations. As the special case of u.e.M.c., we also consider i.i.d. samples.

By the definitions of the acceptance probabilities in Algorithm 1, we can find that the acceptance probabilities are positive. In addition, the size m of training samples is finite. By the theory of Markov chain [24], we can conclude that the samples generated in Algorithm 1 is uniformly ergodic Markov chains (u.e.M.c.). Then we present the definition of u.e.M.c. as follows: Let $(\mathcal{Z}, \mathcal{A})$ be a measurable space, and $\{Z_t\}_{t \geq 1}$ be a Markov chain with transition probability measures $P^n(S|z_j)$, $S \in \mathcal{A}$, $z_j \in \mathcal{Z}$. $P^n(S|z_j)$ is defined as $P^n(S|z_j) = P\{z_{j+n} \in S | Z_i, i < j, Z_j = z_j\}$.

For any $S \in \mathcal{A}$, $z_j \in \mathcal{Z}$, if the transition probability $P^n(S|z_j)$ satisfies $P^n(S|z_j) = P\{z_{j+n} \in S | Z_j = z_j\}$, which is the so called the Markov property of $\{Z_t\}_{t \geq 1}$. This property indicates that given the current state z_j , the past state $z_i, i < j$ is independent of the future state z_{n+j} . By these notations, the definition of u.e.M.c. is given as follows [23].

Definition 1. A Markov chain $\{Z_t\}_{t \geq 1}$ is uniformly ergodic, if for some $\beta < 1, \varphi < \infty$, $\|P^n(\cdot|z) - \varpi(\cdot)\|_{TV} \leq \varphi\beta^n$, for all $n = 1, 2, \dots$, where $\varpi(\cdot)$ is the stationary distribution of $\{Z_t\}_{t \geq 1}$, $\|\cdot\|_{TV}$ is the total variation distance, which is defined as $\|\mu_1 - \mu_2\|_{TV} = \sup_{S \in \mathcal{A}} |\mu_1(S) - \mu_2(S)|$ for two measures μ_1, μ_2 defined on the space $(\mathcal{Z}, \mathcal{A})$.

To measure the generalization ability of MK-MSVCR-TSL algorithm, we should bound the excess generalization error $\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)$, where f_B is the minimizer of $\mathcal{E}(f)$ for all measurable function f . The corresponding best classifier ϕ_B is the Bayes rule, $\phi_B := \arg \min_{j \in Y} \sum_{y \in Y} P_y(x) \cdot \mathbf{1}_{\{y \neq j\}}, x \in X$. To estimate $\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)$, we also present the following some definitions and assumptions.

Definition 2. [29] We call the function f_B is approximated if there exists a constant C_q with an exponent $0 < q \leq 1$ such that $D(\lambda)$ satisfies $D(\lambda) \leq C_q \lambda^q$ for any $\lambda > 0$.

For simplicity, we take $C_q = 1$ in this paper. Since the minimization (3) is taken for the discrete quantity $\mathcal{E}(f)$, bound the excess generalization error involves the capacity of $\bar{\mathcal{H}}_K$. In this paper the capacity of function set is managed by the covering number.

Definition 3. For a subset \mathcal{F} of a metric space and $\epsilon > 0$, the covering number $\mathcal{N}(\mathcal{F}, \epsilon)$ is the minimal $n_1 \in \mathbf{N}$ such that there exist n_1 disks with radius ϵ covering \mathcal{F} .

For $R > 0$, $\mathcal{B}_R = \{f \in \bar{\mathcal{H}}_K : \|f\|_K \leq R\}$. It is a subset of $\mathcal{C}(X)$ and the covering number is well defined [31,29,35]. For any $\epsilon > 0$, $\mathcal{N}(\epsilon) = \mathcal{N}(\mathcal{B}_1, \epsilon)$ is expressed as the covering number of \mathcal{B}_1 .

Definition 4. [29] The RKHS $\bar{\mathcal{H}}_K$ is said to have a polynomial complexity exponent $s > 0$ if there is some constant $C_s > 0$ such that $\ln \mathcal{N}(\epsilon) \leq C_s (1/\epsilon)^s, \forall \epsilon \geq 0$.

Assumption 1 For $\{\mathcal{H}_p\}_{p=1}^n$, \mathcal{H}_p is separable with respect to the norm RKHS, and we set $\kappa = \sup_{x \in X} \sqrt{K_p(x, x)} \leq 1$. In this paper, we assume that there exists a constant $M \geq 0$, we have $f_B \leq M$ [12], and $C = \max\{C_1, C_2\} = 1$.

Our main results on the generalization of MK-MSVCR-TSL algorithm are stated as follows.

Theorem 1. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an u.e.M.c. sample. For any $0 < \delta < 1$, with probability at least $1 - \delta$, inequality

$$\begin{aligned} \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq & 3D(\lambda) + \frac{224(\sqrt{D(\lambda)/\lambda} + R)^2 \| \Gamma_0 \|^2 \ln(\frac{2}{\delta})}{m} \\ & + 2 \left(\frac{448 \| \Gamma_0 \|^2 C_s R^{s+2}}{m} \right)^{\frac{1}{s+1}} \end{aligned}$$

is valid provided that $m \geq 448R \| \Gamma_0 \|^2 \ln(2/\delta) (\ln(2/\delta)/C_s)^{1/s}$.

Corollary 1. Let $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ be an u.e.M.c. sample. Taking $\lambda = (\frac{1}{m})^{\frac{1}{1+s}}$, we have that for any $0 < \delta < 1$, the following inequality is valid with probability at least $1 - \delta$,

$$\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq \bar{C} \left(\frac{1}{m} \right)^{\frac{1}{1+s}},$$

where $\bar{C} = 896 \| \Gamma_0 \|^2 R^2 (4C_s^{\frac{1}{s+1}} + 4 \ln(2/\delta) + 3)$.

Theorem 2. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an i.i.d. sample. For any $0 < \delta < 1$, with probability at least $1 - \delta$, the inequality

$$\begin{aligned} \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq & D(\lambda) \left(2 + \frac{14 \ln(\frac{2}{\delta})}{m\lambda} \right) + \frac{14R^2 \ln(\frac{2}{\delta})}{m} \\ & + \frac{\ln(\frac{2}{\delta})}{m} + 2 \left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}} \end{aligned}$$

is valid provided that $m \geq 74R \ln(2/\delta) (\ln(2/\delta)/C_s)^{1/s}$.

Corollary 2. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an i.i.d. sample. Let $\lambda = (\frac{1}{m})^{\frac{1}{1+s}}$. For any $0 < \delta < 1$, we have that the inequality

$$\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq \hat{C} \left(\frac{1}{m} \right)^{\frac{1}{1+s}}$$

is valid with probability at least $1 - \delta$, where $\hat{C} = 600R^2 (4C_s^{\frac{1}{s+1}} + \ln(2/\delta))$ is a constant.

Theorems 1-2 and Corollaries 1-2 will be proved in Appendix B. Besides, in order to have better showing these theoretical results above, we give the following remarks.

Remark 2. By Corollaries 1-2, we have that $\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \rightarrow 0$, as $m \rightarrow \infty$. This means that the MK-MSVCR algorithm based u.e.M.c. (or i.i.d.) samples is consistent. To our knowledge, all these results above are the first works on this topic.

5. Numerical Studies

In this section, We compare Algorithm 1 with the support vector classification regression machine for multi-class classification (MSVCR) [1], the MSVCR algorithm with multiple kernel learning (MK-MSVCR), the mean weighted multiple kernel SVM [13] with OVO method (MKSVM-OVO), the mean weighted multiple kernel SVM [13] with OVA method (MKSVM-OVA). MKSVM-OVO is a algorithm that the mean weighted multiple kernel support vector machine is combined with one-versus-one strategy to handle multi-class classification, and MKSVM-OVA is a algorithm that the mean weighted multiple kernel support vector machine is combined with one-versus-rest strategy to handle multi-class classification.

5.1. Datasets and experimental setup

We use 9 public datasets from UCI⁵ datasets: Connect4, Postures, Swarm, Twitter, Pavia, TV_News, Mnist, Proyecto, Kegg. For each dataset, we first divide randomly each data sets into the training set D_{train} and the test set D_{test} . The information of these datasets is summarized in Table 1. All experiments were run on Intel 2.80GHz E5-1603 v4 CPU with MATLAB 2018a.

Table 1. 9 Public Datasets

Dataset	$\#D_{train}$	$\#D_{test}$	$\#Input\ Dimension$	$\#Class$
Connect4	50668	16889	126	3
Postures	58571	19524	36	5
Swarm	54036	18012	200	6
Twitter	169850	56616	77	9
Pavia	120000	28152	102	9
TV_News	97263	32422	132	10
Mnist	45000	15000	780	10
Proyecto	47010	15670	7	21
Kegg	49152	16384	25	25

All the experimental results are based on 50 times repeated experiments and the following 8 kernels: a linear kernel $K(a, b) = a'b$, three polynomial kernels $K(a, b) = (1 + a'b)^d$ with $d = \{2, 3, 4\}$ and four RBF kernels $K(a, b) = \exp(-\|a - b\|^2/2\sigma)$, where σ is chosen from the set $\{0.001, 0.01, 1, 10\}$. The other parameters of algorithms are also obtained through 10-fold cross-validation from $[10^{-3}, 10^{-2}, \dots, 10^3]$. For experimental results of classical MSVCR algorithm with single kernel, we choose the best results between among 8 kernels.

Now, we state our experimental process as follows: (i) Train \mathbf{z} and obtain a classifier $\phi_{\mathbf{z}}$ by Algorithm 1. (ii) Test the classifier on the given testing set and calculate the corresponding misclassification rates. (iii) Do process (i)-(ii) above under the same samples \mathbf{z} by the above 4 multi-class classification algorithms, respectively. (iv) Repeat process

⁵ <http://archive.ics.uci.edu/ml/datasets.html>

(ii)-(iii) above for 50 times and calculate the average misclassification rates of 5 algorithms, respectively. In this paper, we use “MSVCR”, “MK-MSVCR”, “MK-OVO” and “MK-OVA” to refer the experimental results of MSVCR, MK-MSVCR, MKSVM-OVO and MKSVM-OVA algorithms based on randomly independent samples, respectively.

5.2. Comparison of misclassification rates

We show the average misclassification rates of 5 algorithms in Tables 2-3.

Table 2. Average misclassification rates (%) with $m = 9000$

Dataset	MK-MSVCR-TSL	MK-MSVCR	MSVCR	MK-OVO	MK-OVA
Connect4	30.04±0.39	30.90±0.42	33.59±0.40	32.39±0.42	33.59±0.48
Postures	21.59±0.32	23.49±0.35	28.02±0.36	24.07±0.39	26.49±0.39
Swarm	35.13±0.36	36.93±0.38	39.15±0.41	38.45±0.47	39.48±0.49
Twitter	11.73±0.29	15.17±0.31	20.21±0.35	17.18±0.37	16.50±0.40
Pavia	20.97±1.09	22.78±1.61	29.18±1.61	27.36±1.01	26.52±1.40
TV_News	30.43±0.31	33.96±0.30	35.23±0.35	33.42±0.31	33.50±0.31
Mnist	11.02±0.32	13.96±0.41	16.66±0.35	16.06±0.51	15.54±0.42
Proyecto	33.44±0.31	34.93±0.35	36.13±0.29	35.43±0.39	35.45±0.45
Kegg	33.95±0.34	35.11±0.30	38.11±0.38	36.30±0.53	36.89±0.47

Table 3. Average misclassification rates (%) with $m = 25000$

Dataset	MK-MSVCR-TSL	MK-MSVCR	MSVCR	MK-OVO	MK-OVA
Connect4	28.01±0.33	29.91±0.35	31.62±0.35	31.49±0.50	31.85±0.45
Postures	18.90±0.34	25.27±0.33	28.87±0.39	27.51±0.41	29.40±0.40
Swarm	33.11±0.32	35.92±0.34	38.16±0.41	36.51±0.51	36.53±0.34
Twitter	10.01±0.25	16.37±0.31	21.21±0.36	20.40±0.33	17.94±0.40
Pavia	19.90±1.01	26.15±1.55	30.01±1.44	32.58±0.84	32.61±1.50
TV_News	28.43±0.32	31.92±0.35	34.25±0.34	31.45±0.39	32.49±0.32
Mnist	10.98±0.34	13.86±0.37	16.06±0.39	14.58±0.34	15.47±0.36
Proyecto	31.50±0.34	34.92±0.35	35.15±0.30	34.38±0.43	34.36±0.45
Kegg	31.44±0.29	33.90±0.35	35.24±0.36	34.44±0.42	34.53±0.43

By Tables 2-3, we can find that for $m = 9000$ (or $m = 25000$), the means of misclassification rates of the proposed MK-MSVCR-TSL algorithm are less than that of other multi-class classification algorithms, and the standard deviations of misclassification rates for the proposed MK-MSVCR-TSL algorithm are also less than that of other multi-class classification algorithms. In detail, according to the experimental results of MSVCR and MK-MSVCR algorithms with randomly independent samples, we can find that the means of misclassification rates of the proposed MK-MSVCR algorithm are less than that of

MSVCR algorithms, it is imply that for more complex and larger multi-classification data, multiple kernel learning can effectively improve the classification accuracy.

Table 4. Wilcoxon tests of average misclassification rates for 5 algorithms

Comparison	R^+	R^-	Hypothesis($\alpha = 0.05$)	Selected
MK-MSVCR-TSL vs. MK-MSVCR	0	45	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MSVCR	0	45	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MK-OVO	0	45	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MK-OVA	0	45	Rejected	MK-MSVCR-TSL

In Table 4, we apply the Wilcoxon signed-rank test ($\alpha = 0.05$)[?] to verify whether there exist statistical significance between the proposed MK-MSVCR-TSL algorithm and other 4 algorithms by using the mean of misclassification rates presented in Table 3. By Table 4, we can find that the proposed MK-MSVCR-TSL has better performance compared to other 4 multi-classification algorithms.

In order to have a better showing the learning performance of the proposed MK-MSVCR-TSL algorithm more clearer, we present Figures 1-6 to compare 50 times misclassification rates of MK-MSVCR-TSL algorithm with that of other algorithms. Here, we use “red cross”, “blue square”, “green circle” and “magenta asterisk” to denote the misclassification rates of MSVCR, MK-MSVCR, MKSVM-OVO and MKSVM-OVA algorithms based on randomly independent samples, respectively. Here m is the size of training sample, and the number of repeat experiments and the misclassification rates are represented on the horizontal axis and the vertical axis, respectively.

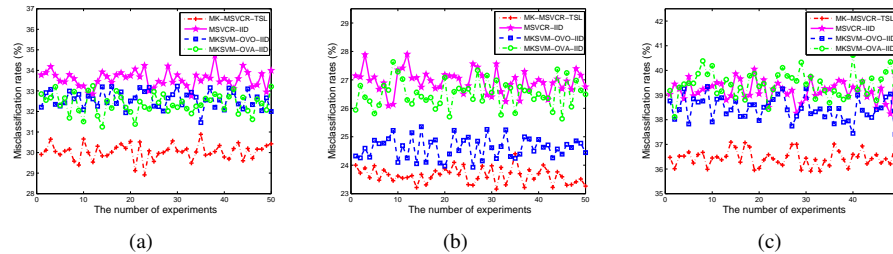


Fig. 1. 50 times experimental misclassification rates for $m = 10000$: (a) Connect4; (b) Postures; (c) Swarm

By Figures 1-6, we can find that for the same size ($m = 10000$ or $m = 20000$) of training sample, all the 50 times misclassification rates of MK-MSVCR-TSL are generally smaller than that of other multi-class classification algorithms. This means that in terms of classification accuracy, our algorithm has obvious advantages. And we can find that as the sample size increases, the advantages of our proposed algorithm are more obvious.

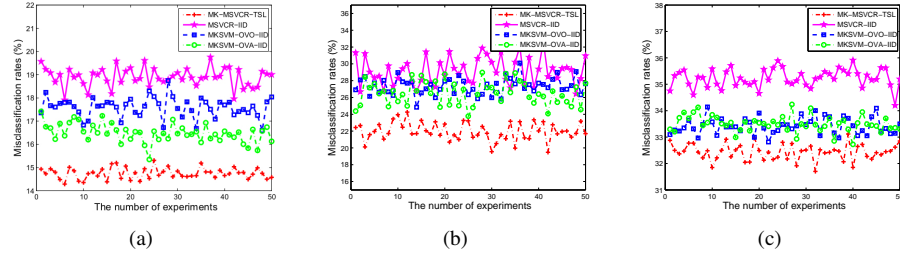


Fig. 2. 50 times experimental misclassification rates for $m = 10000$: (a) Twitter; (b) Pavia; (c) TV_News.

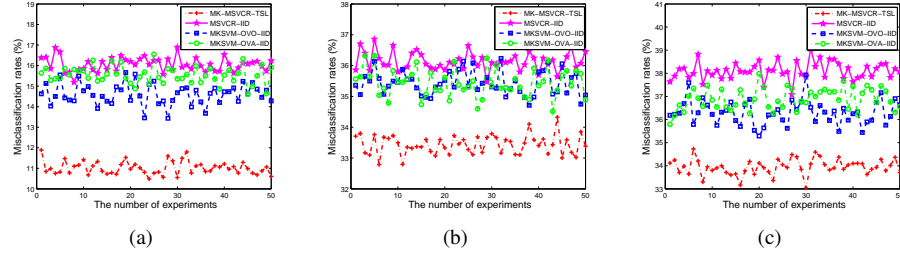


Fig. 3. 50 times experimental misclassification rates for $m = 10000$: (a) Mnist; (b) Projecto; (c) Kegg

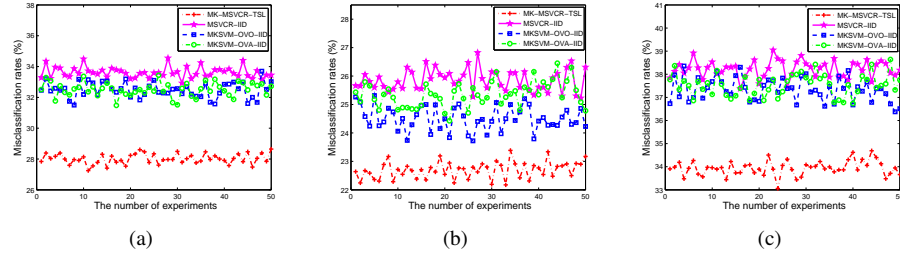


Fig. 4. 50 times experimental misclassification rates for $m = 20000$: (a) Connect4; (b) Postures; (c) Swarm

5.3. Comparison of total time

We show the sampling and training total time of 5 algorithms in Tables 5-6.

By Tables 5-6, we can find that for $m = 9000$ (or $m = 25000$), the sampling and training total time of the proposed MK-MSVCR-TSL is shorter than that of other 4 multi-classification algorithms. Combined with Tables 2-3, for the experimental results of MK-MSVCR-TSL and MK-MSVCR algorithms, we can find that the sampling and training total time of the proposed MK-MSVCR-TSL algorithm is shorter than that of

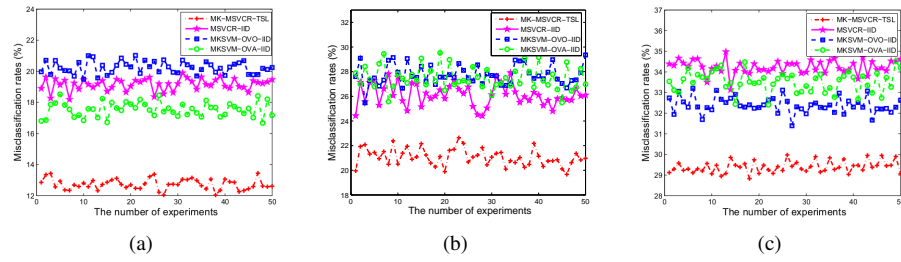


Fig. 5. 50 times experimental misclassification rates for $m = 20000$: (a) Twitter; (b) Pavia; (c) TV_News

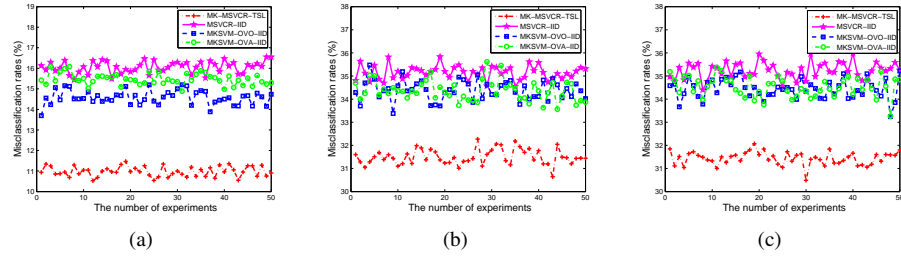


Fig. 6. 50 times experimental misclassification rates for $m = 20000$: (a) Mnist; (b) Projecto; (c) Kegg

Table 5. Sampling and training total time (s) for $m = 9000$

Dataset	MK-MSVCR-TSL	MK-MSVCR	MSVCR	MK-OVO	MK-OVA
Connect4	3834.00	8270.45	5264.87	9930.60	7305.26
Postures	8053.02	23431.51	8914.51	18711.38	15302.13
Swarm	3809.33	9610.83	4088.25	10514.35	8484.36
Twitter	1543.39	5027.24	2532.39	4739.51	4243.14
Pavia	3163.18	9340.25	5231.44	8631.60	8369.17
TV_News	2038.28	5349.02	2454.38	4936.21	3696.07
Mnist	913.72	5381.88	1865.35	5691.39	8170.49
Projecto	2935.07	8138.91	3519.49	9287.14	8036.25
Kegg	3836.14	13796.48	5103.23	12364.96	13505.16
Sum of Time	30126.14	88346.59	38973.91	84807.14	77112.03

MK-MSVCR algorithms, it is imply that for more complex and larger multi-classification data, our algorithm has obvious advantages in terms of sampling and training total time while ensuring certain classification accuracy.

In Table 7, we apply the Wilcoxon signed-rank test ($\alpha = 0.05$)[?] to verify whether there exist statistical significance between the proposed MK-MSVCR-TSL algorithm and other 4 algorithms by using the sampling and training total time presented in Table ??.

Table 6. Sampling and training total time (s) for $m = 25000$

Dataset	MK-MSVCR-TSL	MK-MSVCR	MSVCR	MK-OVO	MK-OVA
Connect4	5009.91	10303.13	7041.91	12343.38	11069.20
Postures	11819.79	43427.13	13434.19	40741.38	36957.31
Swarm	9153.73	23429.15	13007.13	31245.58	29894.29
Twitter	3968.61	15353.13	7872.34	10013.18	8943.33
Pavia	8468.84	20311.17	12935.34	38031.12	35085.71
TV_News	2351.95	7360.40	5340.13	7671.83	7052.36
Mnist	2368.15	8689.35	3399.91	9400.62	10261.12
Proyecto	6054.01	12437.59	8695.15	14343.83	12118.36
Kegg	9846.13	30971.74	19678.04	35083.51	29836.58
Sum of Time	59041.13	172282.79	91404.15	198874.44	181218.27

By Table 7, we can find that the proposed MK-MSVCR-TSL has better performance compared to other 4 multi-classification algorithms.

Table 7. Wilcoxon tests of sampling and training total time for 5 algorithms

Comparison	R^+	R^-	Hypothesis($\alpha = 0.05$)	Selected
MK-MSVCR-TSL vs. MK-MSVCR	0	55	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MSVCR	0	55	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MK-OVO	0	55	Rejected	MK-MSVCR-TSL
MK-MSVCR-TSL vs. MK-OVA	0	55	Rejected	MK-MSVCR-TSL

6. Conclusions

In this paper we firstly considered the generalization bounds of MSVCR algorithm with multiple kernels based on u.e.M.c. samples. As its application, we also established the generalization bounds of MK-MSVCR algorithm for the case of i.i.d. samples and obtained the fast learning rate of MK-MSVCR algorithm for u.e.M.c. and i.i.d. samples, respectively. In addition, we also introduced a new algorithm, the MK-MSVCR-TSL, and showed that the experimental results of the proposed algorithm for public datasets. The experimental results shown that the means of misclassification rates of the MK-MSVCR-TSL and MK-MSVCR are smaller than the classical MSVCR introduced in [1], which implies that the proposed multiple kernel learning can obviously improve the learning performance of the classical MSVCR for large sample size. The experimental results also shown that not only the means of misclassification rates of the MK-MSVCR-TSL are smaller than other multi-class classification algorithms, but also the sampling and training total time of the MK-MSVCR is less than that of other multi-class classification algorithms, which implies that the proposed MK-MSVCR-TSL have obvious competitive strength for learning performance. In other words, the two-stage learning from the given

datasets is a new strategy of improving the learning performance of the classical MSVCR algorithm. Moreover, the experiments display that the proposed algorithm is valid and competitive compared to other multiple kernels multi-class classification methods.

Based on the existing work, there are still several open issues worth further research. For example, applying our method to deep neural networks, using the idea of distributed or parallel to accelerate our method. These problems mentioned above are under our current investigation.

Appendix A

In this section, we give the proof of the main results.

Proposition 1. *Let $f_\lambda = \arg \min_{f \in \mathcal{H}_K} \{\lambda \|f\|_K^2 + \mathcal{E}(f)\}$, $D(\lambda) = \mathcal{E}(f_\lambda) - \mathcal{E}(f_B) + \lambda \|f_\lambda\|_K^2$. For any $\mathbf{z} \in Z^m$, $\lambda \|f_\mathbf{z}\|_K^2 \geq 0$, we have that inequality*

$$\mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_B) \leq \{T_1 + T_2\} + D(\lambda),$$

is valid, where

$$\begin{aligned} T_1 &:= \mathcal{E}(f_\mathbf{z}) - \mathcal{E}_\mathbf{z}(f_\mathbf{z}) - \mathcal{E}(f_B) + \mathcal{E}_\mathbf{z}(f_B), \\ T_2 &:= \mathcal{E}_\mathbf{z}(f_\lambda) - \mathcal{E}(f_\lambda) - \mathcal{E}_\mathbf{z}(f_B) + \mathcal{E}(f_B). \end{aligned}$$

Proof: Since for any $\mathbf{z} \in Z^m$, $\lambda \|f_\mathbf{z}\|_K^2 \geq 0$, we have the following error decomposition inspired by idea from [30],

$$\begin{aligned} \mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_B) &\leq \mathcal{E}(f_\mathbf{z}) - \mathcal{E}(f_B) + \lambda \|f_\mathbf{z}\|_K^2 \\ &= \{\mathcal{E}(f_\mathbf{z}) - \mathcal{E}_\mathbf{z}(f_\mathbf{z}) + \mathcal{E}_\mathbf{z}(f_\mathbf{z}) - \mathcal{E}(f_B) + \mathcal{E}_\mathbf{z}(f_B) - \mathcal{E}_\mathbf{z}(f_B) \\ &\quad + \mathcal{E}_\mathbf{z}(f_\lambda) - \mathcal{E}_\mathbf{z}(f_\lambda) - \mathcal{E}(f_\lambda) + \mathcal{E}(f_\lambda) + \mathcal{E}(f_B) - \mathcal{E}(f_B)\} \\ &\quad + \{\lambda \|f_\mathbf{z}\|_K^2 - \lambda \|f_\lambda\|_K^2 + \lambda \|f_\lambda\|_K^2\} \\ &= \{\mathcal{E}(f_\mathbf{z}) - \mathcal{E}_\mathbf{z}(f_\mathbf{z}) - \mathcal{E}(f_B) + \mathcal{E}_\mathbf{z}(f_B)\} \\ &\quad + \{\mathcal{E}_\mathbf{z}(f_\lambda) - \mathcal{E}(f_\lambda) - \mathcal{E}_\mathbf{z}(f_B) + \mathcal{E}(f_B)\} \\ &\quad + \{\mathcal{E}_\mathbf{z}(f_\mathbf{z}) - \mathcal{E}_\mathbf{z}(f_\lambda) + \lambda \|f_\mathbf{z}\|_K^2 - \lambda \|f_\lambda\|_K^2\} \\ &\quad + \{\mathcal{E}(f_\lambda) - \mathcal{E}(f_B) + \lambda \|f_\lambda\|_K^2\} \\ &= T_1 + T_2 + T_3 + D(\lambda) \leq T_1 + T_2 + D(\lambda) \end{aligned}$$

The last inequality above is follows from the fact that $T_3 \leq 0$ since by the definiton $f_\mathbf{z}$, we have $\mathcal{E}_\mathbf{z}(f_\mathbf{z}) + \lambda \|f_\mathbf{z}\|_K^2 \leq \mathcal{E}_\mathbf{z}(f_\lambda) + \lambda \|f_\lambda\|_K^2$. $D(\lambda)$ is called the regularizing error, which is independent of the sample \mathbf{z} , but is dependent of the space \mathcal{H}_K . Then we complete the proof of Proposition 1.

To prove our results presented in Section 3, our main tools are as follows.

Lemma 1. [29] *Let ξ be a random variable on a probability space Z with mean $E(\xi)$, variance $\sigma^2(\xi) = \sigma^2$, and satisfying $|\xi(z) - E(\xi)| \leq M_\xi$ for almost all $z \in Z$. Then for all $\varepsilon > 0$,*

$$\mathbb{P}\left\{\frac{1}{m} \sum_{i=1}^m \xi(z_i) - E(\xi) \geq \varepsilon\right\} \leq \exp\left\{-\frac{m\varepsilon^2}{2(\sigma^2 + \frac{1}{3}M_\xi\varepsilon)}\right\}.$$

Lemma 2. [29] Let \mathcal{G} be a set of functions on Z such that for some $c_\rho \geq 0$, $|g - E(g)| \leq B$ almost everywhere and $E(g^2) \leq c_\rho E(g)$ for each $g \in \mathcal{G}$. Then for every $\varepsilon > 0$ and $0 < \alpha \leq 1$,

$$\mathbb{P}\left\{\sup_{g \in \mathcal{G}} \frac{E(g) - \frac{1}{m} \sum_{i=1}^m g(z_i)}{\sqrt{E(g) + \varepsilon}} \geq 4\alpha\sqrt{\varepsilon}\right\} \leq \mathcal{N}(\mathcal{G}, \alpha\varepsilon) \exp\left\{-\frac{\alpha^2 m \varepsilon}{2c_\rho + \frac{2}{3}B}\right\}.$$

Lemma 3. [31] Let \mathcal{G} be a countable class of bounded measurable functions, and Z be a u.e.M.c. sample set. Assume that $0 \leq g(z) \leq C_{\mathcal{G}}$ for any $g \in \mathcal{G}$ and for any $z \in Z$. Then for any $\varepsilon > 0$, we have

$$\mathbb{P}\left\{\frac{\frac{1}{m} \sum_{i=1}^m g(z_i) - E(g)}{\sqrt{E(g) + \varepsilon}} \geq \sqrt{\varepsilon}\right\} \leq \exp\left\{\frac{-m\varepsilon}{56C_{\mathcal{G}}\|\Gamma_0\|^2}\right\},$$

where $\|\Gamma_0\| = \sqrt{2}/(1 - \beta_1^{1/2n_1})$, and β_1, n_1 are two positive constants independent of m .

Lemma 4. With all notations as that in Lemma 3, then for $\forall \varepsilon > 0$, we have

$$\mathbb{P}\left\{\sup_{g \in \mathcal{G}} \frac{\frac{1}{m} \sum_{i=1}^m g(z_i) - E(g)}{\sqrt{E(g) + \varepsilon}} \geq 4\sqrt{\varepsilon}\right\} \leq \exp \mathcal{N}(\mathcal{G}, \varepsilon) \left\{\frac{-m\varepsilon}{56C_{\mathcal{G}}\|\Gamma_0\|^2}\right\}.$$

Lemma 5. [6] Let $c_1, c_2 > 0$, and $p_1 > p_2 > 0$. Then, the equation $x^{p_1} - c_1 x^{p_2} - c_2 = 0$ has a unique positive zero x^* . In addition, we have $x^* \leq \max\{(2c_1)^{1/(p_1-p_2)}, (2c_2)^{1/p_1}\}$.

Proposition 2. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an u.e.M.c. sample. For any $0 < \delta < 1$, the following inequality is valid with confidence at least $1 - \delta/2$,

$$T_1 \leq \frac{1}{2}[\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)] + \varepsilon^*(m, \delta/2),$$

where $\varepsilon^*(m, \frac{\delta}{2}) = \max\left\{\frac{448\|\Gamma_0\|^2 R^2 \ln(\frac{2}{\delta})}{m}, \left(\frac{448\|\Gamma_0\|^2 C_s R^{s+2}}{m}\right)^{\frac{1}{s+1}}\right\}$.

Proof: Set $\xi_1 = V(y, f) - V(y, f_B)$. It is clear that ξ_1 varies among a set of functions in accordance with the varying sample \mathbf{z} . Let $\mathcal{G}_R = \{g|g(\mathbf{z}) := V(y, f) - V(y, f_B), f \in \mathcal{B}_R\}$. We have

$$\begin{aligned} E(g) &= \mathcal{E}(f) - \mathcal{E}(f_B) \geq 0, \quad \frac{1}{m} \sum_{i=1}^m g(z_i) = \mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_B), \\ g(z) &= C_1[(1 - yf(x))_+ - (1 - yf_B(x))_+] \cdot \mathbf{1}_{\{y \neq 0\}} \\ &\quad + C_2[(f(x) - f_B(x))(f(x) + f_B(x))] \cdot \mathbf{1}_{\{y=0\}}. \end{aligned}$$

Since $\|f\|_\infty \leq \|f\|_K \leq R$ and $|f_B(x)| \leq M$ almost everywhere, by the restriction $M \leq R$ and $C = \max\{C_1, C_2\} = 1$, we have

$$|g(z)| \leq C_1(R + M) + C_2(R + M)(R + M) \leq 4R^2.$$

It follows that $|g(z) - E(g)| \leq 8R^2$ almost everywhere, and

$$\begin{aligned}
g^2 &= [C_1((1 - yf(x))_+ - (1 - yf_B(x))_+) \cdot \mathbf{1}_{\{y \neq 0\}} \\
&\quad + C_2(f^2(x) - f_B^2(x)) \cdot \mathbf{1}_{\{y=0\}}]^2 \\
&= C_1^2[(1 - yf(x))_+ - (1 - yf_B(x))_+]^2 \cdot \mathbf{1}_{\{y \neq 0\}} \\
&\quad + C_2^2[f^2(x) - f_B^2(x)]^2 \cdot \mathbf{1}_{\{y=0\}} \\
&\leq CC_1[(1 - yf(x))_+ - (1 - yf_B(x))_+](R + M) \cdot \mathbf{1}_{\{y \neq 0\}} \\
&\quad + CC_2[f^2(x) - f_B^2(x)](R^2 + M^2) \cdot \mathbf{1}_{\{y=0\}} \\
&\leq 2R^2 \cdot [C_1((1 - yf(x))_+ - (1 - yf_B(x))_+) \cdot \mathbf{1}_{\{y \neq 0\}} \\
&\quad + C_2(f^2(x) - f_B^2(x)) \cdot \mathbf{1}_{\{y=0\}}].
\end{aligned}$$

Thus $E(g^2) \leq 2R^2E(g)$, and

$$\sup_{f \in \mathcal{B}_R} \frac{\mathcal{E}(f) - \mathcal{E}(f_B) - (\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_B))}{\sqrt{\mathcal{E}(f) - \mathcal{E}(f_B) + \varepsilon}} = \sup_{g \in \mathcal{G}_R} \frac{E(g) - \frac{1}{m} \sum_{i=1}^m g(z_i)}{\sqrt{E(g) + \varepsilon}}.$$

Applying Lemma 4 to the function set \mathcal{G}_R , we have that inequality

$$\sup_{f \in \mathcal{B}_R} \frac{\mathcal{E}(f) - \mathcal{E}(f_B) - (\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_B))}{\sqrt{\mathcal{E}(f) - \mathcal{E}(f_B) + \varepsilon}} = \sup_{g \in \mathcal{G}_R} \frac{E(g) - \frac{1}{m} \sum_{i=1}^m g(z_i)}{\sqrt{E(g) + \varepsilon}} \leq \sqrt{\varepsilon}$$

holds with probability at least $1 - \mathcal{N}(\mathcal{G}_R, \varepsilon) \exp\left\{-\frac{m\varepsilon}{56 \cdot 4R^2 \cdot \|\Gamma_0\|^2}\right\}$.

By Definition 4, we have

$$\begin{aligned}
&\mathbb{P}\left\{\sup_{f \in \mathcal{B}_R} \frac{\mathcal{E}(f) - \mathcal{E}(f_B) - (\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_B))}{\sqrt{\mathcal{E}(f) - \mathcal{E}(f_B) + \varepsilon}} \geq \sqrt{\varepsilon}\right\} \\
&\leq \mathcal{N}(\mathcal{G}_R, \varepsilon) \exp\left\{-\frac{m\varepsilon}{224R^2\|\Gamma_0\|^2}\right\} \\
&\leq \exp\left\{C_s\left(\frac{R}{\varepsilon}\right)^s - \frac{m\varepsilon}{224R^2\|\Gamma_0\|^2}\right\}.
\end{aligned}$$

Let $\delta = \exp\left\{C_s\left(\frac{R}{\varepsilon}\right)^s - \frac{m\varepsilon}{224R^2\|\Gamma_0\|^2}\right\}$. Solving this equation with respect to ε , by Lemma 5, we have

$$\varepsilon = \varepsilon^*(m, \delta) = \max\left\{\frac{448R^2\|\Gamma_0\|^2 \ln(\frac{1}{\delta})}{m}, \left(\frac{448\|\Gamma_0\|^2 C_s R^{s+2}}{m}\right)^{\frac{1}{s+1}}\right\}.$$

It follows that the following inequality holds with the probability at least $1 - \delta$

$$\mathcal{E}(f) - \mathcal{E}(f_B) - (\mathcal{E}_{\mathbf{z}}(f) - \mathcal{E}_{\mathbf{z}}(f_B)) \leq \frac{1}{2}[\mathcal{E}(f) - \mathcal{E}(f_B)] + \varepsilon^*(m, \delta).$$

Replacing f by $f_{\mathbf{z}}$, we have that the following inequality

$$T_1 = \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) - (\mathcal{E}_{\mathbf{z}}(f_{\mathbf{z}}) - \mathcal{E}_{\mathbf{z}}(f_B)) \leq \frac{1}{2}[\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)] + \varepsilon^*(m, \delta/2)$$

is valid with probability at least $1 - \frac{\delta}{2}$. Therefore, we complete the proof of Proposition 2.

By making use of the similar proof method as that in Proposition 2, and Lemma 2, we have

Proposition 3. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an i.i.d. sample. For any $0 < \delta < 1$, we have that the following inequality holds with confidence at least $1 - \delta/2$,

$$T_1 \leq \frac{1}{2}[\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)] + \bar{\varepsilon}(m, \delta/2),$$

where $\bar{\varepsilon}(m, 2/\delta) = \max \left\{ \frac{300R^2 \ln(2/\delta)}{m}, \left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}} \right\}$.

Proposition 4. Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is a u.e.M.c. sample. For any $0 < \delta < 1$, we have that the following inequality holds with the probability at least $1 - \delta/2$,

$$T_2 \leq \frac{1}{2}D(\lambda) + \frac{112(\sqrt{D(\lambda)/\lambda} + R)^2 \cdot \|\Gamma_0\|^2 \ln(\frac{2}{\delta})}{m}.$$

Proof: By the definitions of f_λ and $D(\lambda)$, we have $\lambda \|f_\lambda\|_K^2 \leq \mathcal{E}(f_\lambda) - \mathcal{E}(f_B) + \lambda \|f_\lambda\|_K^2 = D(\lambda)$. It follows that $\|f_\lambda\|_\infty \leq \|f_\lambda\|_K \leq \sqrt{D(\lambda)/\lambda}$. Set $\xi_2 = V(y, f_\lambda) - V(y, f_B)$, we have

$$\begin{aligned} \xi_2 &= C_1(1 - yf_\lambda(x))_+ \cdot \mathbf{1}_{\{y \neq 0\}} + C_2(y - f_\lambda(x))^2 \cdot \mathbf{1}_{\{y=0\}} \\ &\quad - C_1(1 - yf_B(x))_+ \cdot \mathbf{1}_{\{y \neq 0\}} - C_2(y - f_B(x))^2 \cdot \mathbf{1}_{\{y=0\}} \\ &= C_1[(1 - yf_\lambda(x))_+ - (1 - yf_B(x))_+] \cdot \mathbf{1}_{\{y \neq 0\}} + C_2[f_\lambda^2(x) - f_B^2(x)] \cdot \mathbf{1}_{\{y=0\}}, \end{aligned}$$

then $T_2 = \frac{1}{m} \sum_{i=1}^m \xi_2(z_i) - E(\xi_2)$. Since $|f_B| \leq M \leq R$ almost everywhere, we have

$$\begin{aligned} |\xi_2| &\leq |C_1[(1 - yf_\lambda(x))_+ - (1 - yf_B(x))_+] \cdot \mathbf{1}_{\{y \neq 0\}} \\ &\quad + C_2[(f_\lambda(x) - f_B(x))(f_\lambda(x) + f_B(x))] \cdot \mathbf{1}_{\{y=0\}}| \\ &\leq C(\sqrt{D(\lambda)/\lambda} + R) + C(\sqrt{D(\lambda)/\lambda} + R)(\sqrt{D(\lambda)/\lambda} + R) \\ &\leq 2(\sqrt{D(\lambda)/\lambda} + R)^2 := 2b. \end{aligned}$$

Hence $|\xi_2 - E(\xi_2)| \leq M_{\xi_2} := 4b$, $|\xi_2| \leq 2(\sqrt{D(\lambda)/\lambda} + R)^2 := 2b$. Moreover, we have

$$\begin{aligned} E(\xi_2^2) &= E[C_1((1 - yf_\lambda(x))_+ - (1 - yf_B(x))_+) \cdot \mathbf{1}_{\{y \neq 0\}} \\ &\quad + C_2(f_\lambda(x)^2 - f_B(x)^2) \cdot \mathbf{1}_{\{y=0\}}]^2 \\ &= E\{C_1[(1 - yf_\lambda(x))_+ - (1 - yf_B(x))_+] \cdot \mathbf{1}_{\{y \neq 0\}}\}^2 \\ &\quad + E\{C_2[(f_\lambda(x) - f_B(x))(f_\lambda(x) + f_B(x))] \cdot \mathbf{1}_{\{y=0\}}\}^2 \\ &\leq C^2 \|f_\lambda(x) - f_B(x)\|_\rho^2 + C^2 \|f_\lambda(x) - f_B(x)\|_\rho^2 (\sqrt{D(\lambda)/\lambda} + R)^2 \\ &\leq C^2 (\|f_\lambda(x) - f_B(x)\|_\rho^2 + \lambda \|f_\lambda\|_K^2) + C^2 (\|f_\lambda(x) - f_B(x)\|_\rho^2 \\ &\quad + \lambda \|f_\lambda\|_K^2) (\sqrt{D(\lambda)/\lambda} + R)^2 \\ &\leq C^2 D(\lambda) + C^2 D(\lambda) (\sqrt{D(\lambda)/\lambda} + R)^2 \\ &= D(\lambda) (1 + (\sqrt{D(\lambda)/\lambda} + R)^2) = D(\lambda) (1 + b). \end{aligned}$$

Applying Lemma 3, we have that for any $\varepsilon > 0$,

$$\mathbb{P} \left\{ \frac{\frac{1}{m} \sum_{i=1}^m \xi_2(z_i) - E(\xi_2)}{\sqrt{E(\xi_2)} + \varepsilon} \geq \sqrt{\varepsilon} \right\} \leq \exp \left\{ \frac{-m\varepsilon}{56 \cdot \|\Gamma_0\|^2 \cdot 2b} \right\}.$$

It follows that for any $0 < \delta < 1$, with probability at least $1 - \delta$, inequality

$$\frac{1}{m} \sum_{i=1}^m \xi_2(z_i) - E(\xi_2) \leq \frac{1}{2} D(\lambda) + \frac{112(\sqrt{D(\lambda)/\lambda} + R)^2 \cdot \|T_0\|^2 \ln(\frac{2}{\delta})}{m}$$

is valid. Then we accomplish the proof of Proposition 4.

Similar to the proof of Proposition 4, we obtain the following bound of T_2 for i.i.d. samples.

Proposition 5. *Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is an i.i.d. sample. For any $0 < \delta < 1$, the following inequality holds with the probability at least $1 - \delta/2$,*

$$T_2 \leq D(\lambda) \left(1 + \frac{7 \ln(\frac{2}{\delta})}{m\lambda}\right) + \frac{7R^2 \ln(\frac{2}{\delta})}{m} + \frac{\frac{1}{2} \ln(\frac{2}{\delta})}{m}.$$

Proof: Applying Lemma 1, by Proposition 4, we have that for any $t > 0$, $\frac{1}{m} \sum_{i=1}^m \xi_2(z_i) - E(\xi_2) \leq t$, with confidence at least

$$\begin{aligned} 1 - \exp\left\{-\frac{mt^2}{2(\sigma^2(\xi_2) + \frac{1}{3}M_{\xi_2}t)}\right\} &\geq 1 - \exp\left\{-\frac{mt^2}{2[D(\lambda)(1+b) + \frac{1}{3} \cdot 4bt]}\right\} \\ &= 1 - \exp\left\{-\frac{mt^2}{2D(\lambda)(1+b) + \frac{8}{3}bt}\right\}. \end{aligned}$$

Select t^* as the only positive solution of the equation

$$-\frac{mt^2}{2D(\lambda)(1+b) + \frac{8}{3}bt} = \ln \delta.$$

So, $\frac{1}{m} \sum_{i=1}^m \xi_2(z_i) - E(\xi_2) \leq t^*$ holds with probability $1 - \delta$. Then

$$\begin{aligned} t^* &= \frac{\frac{4b}{3} \ln(\frac{1}{\delta}) + \sqrt{(\frac{4b}{3} \ln(\frac{1}{\delta}))^2 + 2D(\lambda)(1+b)m \ln(\frac{1}{\delta})}}{m} \\ &\leq \frac{8b \ln(\frac{1}{\delta})}{3m} + \sqrt{\frac{2D(\lambda)(1+b) \ln(\frac{1}{\delta})}{m}} \\ &\leq \frac{8b \ln(\frac{1}{\delta})}{3m} + D(\lambda) + \frac{(1+b) \ln(\frac{1}{\delta})}{2m}. \end{aligned}$$

Recall $b = (\sqrt{D(\lambda)/\lambda} + R)^2 \leq 2(D(\lambda)/\lambda + R^2)$. It follows that

$$t^* \leq D(\lambda) \left(1 + \frac{7 \ln(\frac{1}{\delta})}{m\lambda}\right) + \frac{7R^2 \ln(\frac{1}{\delta})}{m} + \frac{\frac{1}{2} \ln(\frac{1}{\delta})}{m}.$$

Then we accomplish the proof of Proposition 5.

Appendix B

Proof of Theorem 1: Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is a u.e.M.c. sample. Similar to the proof of Theorem 2, we have that for any $0 < \delta < 1$, with probability at least $1 - \delta$, the inequality

$$\begin{aligned} \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) &\leq 3D(\lambda) + \frac{224 \cdot (\sqrt{D(\lambda)/\lambda} + R)^2 \cdot \|\Gamma_0\|^2 \ln(\frac{2}{\delta})}{m} \\ &\quad + 2 \left(\frac{448 \|\Gamma_0\|^2 C_s R^{s+2}}{m} \right)^{\frac{1}{s+1}} \end{aligned}$$

is valid provided that $m \geq 448R \|\Gamma_0\|^2 \ln(2/\delta) (\ln(2/\delta)/C_s)^{1/s}$. Then, we accomplish the proof of Theorem 1.

Proof of Corollary 1: Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is a u.e.M.c. sample. According to Definition 2, $D(\lambda) \leq \lambda^q$. Then for any $0 < \delta < 1$, with probability at least $1 - \delta$, we have

$$\begin{aligned} &\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \\ &\leq 3D(\lambda) + \frac{224 \cdot (\sqrt{D(\lambda)/\lambda} + R)^2 \cdot \|\Gamma_0\|^2 \ln(\frac{2}{\delta})}{m} + 2\varepsilon^*(m, \delta/2) \\ &\leq 3D(\lambda) + \frac{224 \cdot (\sqrt{D(\lambda)/\lambda} + R)^2 \cdot \|\Gamma_0\|^2 \ln(\frac{2}{\delta})}{m} \\ &\quad + \frac{896 \|\Gamma_0\|^2 R^2 \ln(\frac{2}{\delta})}{m} + 2 \left(\frac{448 \|\Gamma_0\|^2 C_s R^{s+2}}{m} \right)^{\frac{1}{s+1}} \\ &\leq \bar{C} \left(\lambda^q + \frac{\lambda^{q-1}}{m} + \frac{1}{m} + \frac{\lambda^{(q-1)/2}}{m} + \frac{1}{m} + \left(\frac{1}{m} \right)^{\frac{1}{1+s}} \right), \end{aligned}$$

where $\bar{C} = 896 \|\Gamma_0\|^2 R^2 (4C_s^{\frac{1}{s+1}} + 4 \ln(2/\delta) + 3)$.

Let $\lambda = (\frac{1}{m})^{\frac{1}{1+s}}$ and q close to 1, so the inequality

$$\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq \bar{C} \left(\lambda^q + \frac{\lambda^{q-1}}{m} + \frac{1}{m} + \frac{\lambda^{(q-1)/2}}{m} + \frac{1}{m} + \left(\frac{1}{m} \right)^{\frac{1}{1+s}} \right) \leq \bar{C} \left(\frac{1}{m} \right)^{\frac{1}{1+s}}$$

is valid with probability at least $1 - \delta$, where $\bar{C} = 896 \|\Gamma_0\|^2 R^2 (4C_s^{\frac{1}{s+1}} + 4 \ln(2/\delta) + 3)$ is a constant. Then, we finish the proof of Corollary 1.

Proof of Theorem 2: Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is i.i.d. sample. With the bounds of T_1 (Prop.3), T_2 (Prop.5) and $D(\lambda)$ (Def.3), we have that with confidence $1 - \delta$,

$$\begin{aligned} \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) &\leq T_1 + T_2 + D(\lambda) \\ &\leq \frac{1}{2} [\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B)] + D(\lambda) \left(1 + \frac{7 \ln(\frac{2}{\delta})}{m\lambda} \right) + \frac{7R^2 \ln(\frac{2}{\delta})}{m} + \frac{\frac{1}{2} \ln(\frac{2}{\delta})}{m} + \bar{\varepsilon}(m, \delta/2). \end{aligned}$$

For $\bar{\varepsilon}(m, \delta/2)$, the inequality $\left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}} \geq \frac{300R^2 \ln(2/\delta)}{m}$ is valid with $m \geq 74R \ln(2/\delta) (\ln(2/\delta)/C_s)^{1/s}$, we get $\bar{\varepsilon}(m, \delta/2) = \left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}}$. Thus, for any $0 <$

$\delta < 1$, with probability at least $1 - \delta$, we have

$$\begin{aligned} \mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) &\leq D(\lambda) \left(2 + \frac{14 \ln(\frac{2}{\delta})}{m\lambda} \right) + \frac{14R^2 \ln(\frac{2}{\delta})}{m} \\ &\quad + \frac{\ln(\frac{2}{\delta})}{m} + 2 \left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}}. \end{aligned}$$

Then, we finish the proof of Theorem 2.

Proof of Corollary 2: Assume $\mathbf{z} = \{z_i\}_{i=1}^m \in Z^m$ is i.i.d. sample. According to Definition 2, $D(\lambda) \leq \lambda^q$. Then for any $0 < \delta < 1$, with probability at least $1 - \delta$, we have

$$\begin{aligned} &\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \\ &\leq D(\lambda) \left(2 + \frac{14 \ln(\frac{2}{\delta})}{m\lambda} \right) + \frac{14R^2 \ln(\frac{2}{\delta})}{m} + \frac{\ln(\frac{2}{\delta})}{m} + 2\bar{\varepsilon}(m, \delta/2) \\ &\leq \lambda^q \left(2 + \frac{14 \ln(\frac{2}{\delta})}{m\lambda} \right) + \frac{14R^2 \ln(\frac{2}{\delta})}{m} + \frac{\ln(\frac{2}{\delta})}{m} \\ &\quad + \frac{600R^2 \ln(\frac{2}{\delta})}{m} + 2 \left(\frac{300R^2 C_s (4R)^s}{m} \right)^{\frac{1}{s+1}} \\ &\leq \widehat{C} \left(\lambda^q + \frac{\lambda^q}{m\lambda} + \frac{1}{m} + \frac{1}{m} + \frac{1}{m} + \left(\frac{1}{m} \right)^{\frac{1}{1+s}} \right), \end{aligned}$$

where $\widehat{C} = 600R^2(4C_s^{\frac{1}{s+1}} + \ln(2/\delta))$. Let $\lambda = (\frac{1}{m})^{\frac{1}{1+s}}$ and q close to 1, so the inequality

$$\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_B) \leq \widehat{C} \left(\lambda^q + \frac{\lambda^q}{m\lambda} + \frac{3}{m} + \left(\frac{1}{m} \right)^{\frac{1}{1+s}} \right) \leq \widehat{C} \left(\frac{1}{m} \right)^{\frac{1}{1+s}}$$

is valid with probability at least $1 - \delta$, where $\widehat{C} = 600R^2(4C_s^{\frac{1}{s+1}} + \ln(2/\delta))$ is a constant. Then, we accomplish the proof of Corollary 2.

Acknowledgments. This work is supported by Scientific Research Foundation of Hubei University of Education for Talent Introduction (No.ESRC20230008), and Open Foundation of Hubei Key Laboratory of Applied Mathematics (Hubei University) (HBAM202304).

References

1. Angulo, C., Parra, X., Català, A.: K-svc: A support vector machine for multi-class classification. *Neurocomputing* 55(1–2), 57–77 (2003)
2. Bennett, K., Mangasarian, O.L.: Combining support vector and mathematical programming methods for induction. *Advances in Kernel Methods-SV Learning* pp. 307–326 (1999)
3. Bottou, L., Cortes, C., Denker, J.S., Drucker, H., Guyon, I., Jackel, L.D., LeCun, Y., Muller, U.A., Sackinger, E., Simard, P.: Comparison of classifier methods: A case study in handwritten digit recognition. In: *Proceedings of the 12th IAPR International Conference on Pattern Recognition*. pp. 77–82 (1994)
4. Chao, Z.: Machine learning-based intelligent weather modification forecast in smart city potential area. *Computer Science and Information Systems* (20), 631–656 (2023)
5. Chavaltada, C., Pasupa, K., Hardoon, D.R.: Combining multiple features for product categorisation by multiple kernel learning. *International Conference on Computing and Information Technology* pp. 3–12 (2018)

6. Cucker, F., Smale, S.: Best choices for regularization parameters in learning theory: On the bias-variance problem. *Foundations of Computational Mathematics* 2(4), 413–428 (2002)
7. Dong, Z., Gong, J., Zou, B., Wang, Y., Xu, J.: Generalization and learning rate of multi-class support vector classification and regression. *International Journal of Wavelets, Multiresolution and Information Processing* (20), 2250017 (2022)
8. Dong, Z., Qin, Y., Zou, B., Xu, J., Tang, Y.Y.: Lmsvcr: Novel effective method of semi-supervised multi-classification. *Neural Computing and Application* (34), 3857–3873 (2022)
9. Dong, Z., Xu, C., Xu, J., Zou, B., Zeng, J., Tang, Y.Y.: Generalization capacity of multi-class svm based on markovian resampling. *Pattern Recognition* (142), 109720 (2023)
10. Duan, Y., Zou, B., Xu, J., Chen, F., Wei, J., Tang, Y.Y.: Oaa-svm-ms: A fast and efficient multi-class classification algorithm. *Neurocomputing* (454), 448–460 (2021)
11. Duġan, U., Glasmachers, T., Igel, C.: A unified view on multi-class support vector classification. *Journal of Machine Learning Research* 17(45), 1–32 (2016)
12. Feng, Y., Yang, Y., Zhao, Y., Lv, S., Suykens, J.A.: Learning with Kernelized Elastic Net Regularization. KU Leuven, Leuven, Belgium (2014)
13. Gönen, M., Alpaydin, E.: Multiple kernel learning algorithms. *Journal of Machine Learning Research* (12), 2211–2268 (2011)
14. Huang, C.L., Dun, J.F.: A distributed pso-csvm hybrid system with feature selection and parameter optimization. *Applied Soft Computing* 8(4), 1381–1391 (2008)
15. Krebel, U.H.G.: Pairwise classification and support vector machines. *Advances in kernel methods: support vector learning* pp. 255–268 (1999)
16. Lanckriet, G.R.G., Cristianini, N., Bartlett, P.L., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research* (5), 27–72 (2004)
17. Lauriola, I., Gallicchio, C., Aiolli, F.: Enhancing deep neural networks via multiple kernel learning. *Pattern Recognition* (101), 107194 (2020)
18. Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association* 99(465), 67–81 (2004)
19. Lin, S.W., Ying, K.C., Chen, S.C., Lee, Z.J.: Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications* 35(4), 1817–1824 (2008)
20. Luo, J., Orabona, F., Feroni, M., Caputo, B., Cesa-Bianchi, N.: Om-2: An online multi-class multi-kernel learning algorithm. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops* pp. 43–50 (2010)
21. Lv, S.G., Zhou, F.Y.: Optimal learning rates of l^p -type multiple kernel learning under general conditions. *Information Sciences* (294), 255–268 (2015)
22. Lv, S.G., Zhu, J.D.: Error bounds for l^p -norm multiple kernel learning with least square loss. *Abstract and Applied Analysis* pp. 1–18 (2012)
23. Meyn, S.P., Tweedie, R.L.: *Markov Chains and Stochastic Stability*. Springer Science & Business Media (2012)
24. Qian, M., Nie, F., Zhang, C.: Efficient multi-class unlabeled constrained semi-supervised svm. In: *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. pp. 1665–1668 (2009)
25. Vapnik, V.: *Statistical Learning Theory*. Wiley, New York (1998)
26. Wang, T., Su, H., Li, J.: Dws-mkl: Depth-width-scaling multiple kernel learning for data classification. *Neurocomputing* (411), 455–467 (2020)
27. Weston, J., Watkins, C.: Support vector machines for multi-class pattern recognition. *Esann* pp. 219–224 (1999)
28. Wilson, C.M., Li, K.Q., Yu, X.Q., Kuan, P.F., Wang, X.F.: Multiple-kernel learning for genomic data mining and prediction. *BMC Bioinformatics* 20(1), 1–7 (2019)

29. Wu, Q., Ying, Y., Zhou, D.X.: Learning rates of least-square regularized regression. *Foundations of Computational Mathematics* 6(2), 171–192 (2006)
30. Wu, Q., Zhou, D.X.: Svm soft margin classifiers: Linear programming versus quadratic programming. *Neural Computation* 17(5), 1160–1187 (2005)
31. Xu, J., Tang, Y.Y., Zou, B., Xu, Z., Li, L., Lu, Y., Zhang, B.: The generalization ability of svm classification based on markov sampling. *IEEE Transactions on Cybernetics* 45(6), 1169–1179 (2015)
32. Yang, Y., Guo, Y., Chang, X.: Angle-based cost-sensitive multicategory classification. *Computational Statistics & Data Analysis* (156), 107107 (2021)
33. Yao, B., Liu, S., Wang, L.: Using machine learning approach to construct the people flow tracking system for smart cities. *Computer Science and Information Systems* (20), 679–700 (2023)
34. Yi, Z.H., Etemadi, A.H.: Line-to-line fault detection for photovoltaic arrays based on multi-resolution signal decomposition and two-stage support vector machine. *IEEE Transactions on Industrial Electronics* 64(11), 8546–8556 (2017)
35. Zou, B., Li, L., Xu, Z.: The generalization performance of erm algorithm with strongly mixing observations. *Machine Learning* 75(3), 275–295 (2009)

Zijie Dong received the Ph.D. degree from the Faculty of Mathematics and Statistics of Hubei University, China, in 2022. She is currently working at School of Mathematics and Statistics, Hubei University of Education, Wuhan, 430205, China. Her current research interests include statistical learning theory, machine learning, and pattern recognition.

Fen Chen received the Ph.D. degree from the Faculty of Mathematics and Statistics of Hubei University, China, in 2019. She is currently working at School of Finance, Hubei University of Economics, Wuhan, 430205, China.

Yu Zhang received the Ph.D. degree from the Nanjing University of Aeronautics and Astronautics, China, in 2021. He is currently working at School of Mathematics and Statistics, Hubei University of Education, Wuhan, 430205, China.

Received: January 24, 2023; Accepted: December 10, 2023.

An Approach for Supporting Transparent ACID Transactions over Heterogeneous Data Stores in Microservice Architectures

Lazar Nikolić, Vladimir Dimitrieski, and Milan Čeliković

University of Novi Sad, Faculty of Technical Sciences
Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia
{lazar.nikolic,dimitrieski,milancel}@uns.ac.rs

Abstract. Microservice architectures (MSA) are becoming a preferred architectural style for data-driven applications. A transaction within MSA can include remote calls to multiple services, turning it into a distributed transaction. Participating services may have their own data stores running local transactions with varying levels of transactional support and consistency guarantees. Coordinating distributed transactions in such an environment is a key challenge for MSA. The existing approaches are either highly consistent at the expense of scalability or scalable at the expense of consistency. Furthermore, implementing any of them requires architectural and code adaptation. In this article, we present the Service Proxy Transaction Management (SPTM) approach, which offers scalable reads and ACID transactions in MSA. The novelty of this approach is that it is based on intercepting inbound messages to services, rather than having services directly communicate with a transaction manager. As a result, transaction management is completely transparent to services and has little-to-no impact on code or architecture. We provide experimental results showing that SPTM can outperform lock-based approaches by up to a factor of 2, while still providing high consistency without the scaling bottleneck associated with locking.

Keywords: distributed transaction management, consistency, microservice, saga, 2pc, acid, base.

1. Introduction

In the past decade, there has been a shift from traditional, monolithic services to MSA utilizing multiple small services for data-driven applications. The trend has spread to many areas of the software industry, such as smart cities [55,46,10], e-commerce and financial systems [57,40], driving assistance [53], and edge computing [27]. The general idea is to split a complex domain into subdomains and assign each to a service. Ideally, services are small, mostly self-contained, and communicate with other services only when their domain boundary is crossed. This leads to improvements in scalability, robustness, and development flexibility enabled by loose coupling between services [48,45,66]. But MSA also brings new challenges to the table, with data management being one of the major categories [48,45,66]. One notable challenge in this category is facilitating transactions, which arises from the way they are executed in MSA. With a monolithic service, a transaction is comprised of a set of direct function calls to different submodules. The

transaction and the data it handles are under the control of the monolithic service. In contrast, transactions in MSA include inter-service remote calls via messaging protocols such as Hypertext Transfer Protocol (HTTP) or, more commonly, HTTP Secure (HTTPS). The communication in this manner happens at the *service communication level*.

Each service may have its own data store in a database-per-service fashion, with which it communicates at the *database communication level*. A particular data store used by a particular service might be chosen to address the specific needs of the service. For example, a service responsible for connections between people in a social network application would benefit the most from a graph database. This is known as the polyglot persistence paradigm [30], and applying it creates a heterogeneous environment of data stores. Providing a uniform level of consistency and transactional support is difficult even in homogeneous environments; heterogeneity takes it to another level of complexity. Transactional support and consistency levels offered by data stores in a system following the polyglot persistence principle are hardly ever uniform. For example, while relational databases are highly consistent with full transactional support, NoSQL data stores come with low consistency levels and transactional limitations. For instance, popular NoSQL data stores, such as Elasticsearch [16] and Cassandra [17], ensure atomicity only at the level of individual documents.

1.1. Motivation

Even if all the data stores in a system were identical, the challenge of coordinating a distributed transaction is still present. A distributed transaction in MSA is composed of multiple local transactions, each running on a different data store. Data stores have no inherent mechanism to connect local transactions to a larger context of a distributed transaction. Thus, distributed transaction coordination must be built on top of the data stores. Furthermore, transactional properties, such as Atomicity, Consistency, Isolation, and Durability (ACID) guarantees, do not extend beyond local transactions. This means that even if a distributed transaction is composed entirely of local ACID transactions, it is not ACID by default.

Workloads in MSA generally fall under two broad categories: Online Analytical Processing (OLAP) and Online Transactional Processing (OLTP). OLAP is characterized by complex, long-running ad hoc queries over many items. In contrast, OLTP is characterized by short-lived transactions on a small number of items, usually involving write operations. OLTP appears in data-driven applications very often [48]. An example of OLTP is a typical e-commerce scenario: a user checks out a product, product stock is updated, and payment is made. Within MSA, each step may be carried out by a different service in a distributed transaction. ACID guarantees are natural requirements for OLTP workloads due to the high consistency demands of data-driven applications handling them.

The level of consistency of a transaction can be tied to its isolation level [69]. Isolation level is defined by how many *isolation anomalies* [6] it allows to happen: fewer anomalies indicate a higher isolation level. An isolation anomaly is an unwanted effect on data caused by concurrent execution of transactions. An example of an isolation anomaly is *Lost Update* [1], in which a transaction can overwrite the result from another active transaction. Consider an example with transactions $T1$ and $T2$ that attempt to increment value $i=1$. Both $T1$ and $T2$ read the value $i=1$ and write the incremented value, which is $i=2$. The

expected outcome would be $i=3$, but a result from one of the transactions is effectively lost.

Isolation levels are defined in the order from the least to the most restrictive, as follows: *READ UNCOMMITTED*, *READ COMMITED*, *CURSOR STABILITY*, *REPEATABLE READ*, *SNAPSHOT*, and *SERIALIZABLE* [6]. The higher the level, the more checks a system must run. This ultimately means that a higher isolation level leads to higher consistency, but at the cost of lower performance. A high consistency level is also commonly referred to as strong or strict consistency, whereas lower consistency levels are referred to as weak consistency.

Existing approaches to handling distributed transactions are synchronous and strongly consistent, or asynchronous and weakly consistent. The shortcomings of synchronous approaches are attributed to the locking of involved items [67,28], which is absent in asynchronous approaches. The asynchronous approaches in turn bring challenges related to weak consistency, such as transactions reading uncommitted values. There is a space of solutions to be explored that avoids locking, while still offering a high consistency level. Existing approaches also incur changes to both new and ongoing projects. Code and sometimes architectural changes are necessary to accommodate the chosen distributed transaction handling approach. Developers must be familiar with how distributed transactions work within an approach to effectively apply it. This is a notoriously difficult topic, so minimizing this requirement can be a huge boon to the development process. Unfortunately, there is a lack of good and intuitive abstractions to help developers tackle this issue, particularly for strongly consistent approaches, which can be a large barrier to adoption [41,42].

1.2. Contribution

In this article, we describe Service Proxy Transaction Management (SPTM), an approach to distributed transaction management in MSA that provides ACID guarantees for OLTP workloads. ACID is provided even in heterogeneous data store environments, in which not all data stores have ACID capabilities. SPTM combines the lock-free mechanism of asynchronous approaches with the high consistency of synchronous approaches by masking intermediate results of ongoing transactions. SPTM also aims to be as non-invasive to the existing codebase as possible with a low-performance overhead. It does so by intercepting, evaluating, and modifying messages at the service communication level. From the developer's perspective, a distributed transaction within SPTM is a collection of remote service calls with no additional libraries or frameworks, making the approach easy to understand and use. We believe that offering transparent ACID capabilities with the flexibility to combine it with both synchronous and asynchronous messaging makes SPTM very valuable for microservice developers. We also explore the possibilities and limits of the approach in which transactions are managed at the service communication level, using only the information available in the messages.

1.3. Article structure

In addition to the *Introduction* and *Conclusion*, the article comprises five sections. In the *Related work* section, we present and compare existing approaches to SPTM. The SPTM

approach is described in the *Service Proxy Transaction Management — SPTM* section. In the *Evaluation* section, we run benchmarks on an SPTM implementation and compare it to an implementation of another commonly used approach. We also discuss and explain the results of benchmarks. In the *Threats to validity* section, we briefly present factors that could affect the results of this article, primarily the ones presented in the *Evaluation* section. In the *Limitations and future work* section, we describe what the current limitations are and how we plan to overcome them.

2. Related Work

There are several research threads that work on addressing the problem of facilitating distributed OLTP transactions in MSA from various angles. While they are very distinct from each other, every thread exhibits three main characteristics: (i) support for synchronous and asynchronous execution, (ii) consistency guarantees and ACID support, and (iii) impact on the codebase and the software design. In this section, we compare current approaches through the prism of these characteristics. We have found that none fill the niche of both synchronous and asynchronous execution support, high consistency and ACID support, and low impact on the codebase — all of which SPTM strives to offer.

Two-phase commit protocol (2PC) can be considered a good match for OLTP workloads due to its synchronous nature and high consistency. As the name suggests, 2PC carries out transaction execution in two phases: the prepare phase and the commit phase. In the prepare state, participants check if conditions for applying changes. If the check passes, the changes are applied in the commit phase. The phases are coordinated by a transaction coordinator component that sends messages to participants. During the execution of both phases, locks are kept on the items involved in the transaction.

The locking mechanism of 2PC is a bottleneck for scalability and its main downside [67,28]. Furthermore, the 2PC coordinator acts as a single point of failure in the system: new transactions cannot be accepted nor executed if the coordinator is unavailable. 2PC also heavily relies on all data stores providing commands for controlling a transaction flow like *BEGIN*, *COMMIT*, and *ABORT* in SQL. Meeting this requirement is not always possible in heterogeneous data store environments. For these reasons, 2PC is now rarely used in practice within MSA [48]. There are also several 2PC variants, such as Distributed Strong Strict Two-Phase Locking (SS2PL) [8] and 2PC* [28] which scale better, but still fall behind approaches that eschew locking. 2PC only supports synchronous execution with high consistency across all variants. It also comes with code changes and design limitations, as it requires the use of client libraries and narrows the selection of data stores to transactional ones.

Despite falling out of focus of literature and open-source MSA projects in the recent years, 2PC is still present in industry and academia. It is still being used in industry with at least 8-16% practitioners claiming they use it [48], while also being present in research, with some more recent papers still referencing it or using it for comparison [28,73].

Saga is a pattern used for coordinating distributed transactions in Event-Driven Architecture (EDA). It consists of a series of subtransactions carried out by multiple services. Transactions are coordinated by creating an event signaling that the next subtransaction should start. This can be done in two ways [63]:

- **Orchestrated Saga** utilizes a central coordinator that coordinates subtransactions. When a service finishes a subtransaction, it notifies the coordinator. The coordinator in turn creates an event for the next service to start its subtransaction. Orchestrated Sagas help implement complex transactions by having them defined in the coordinator, with all the steps clearly laid out. The main disadvantage is that the coordinator is a potential point of failure. It is also a complex component to develop and maintain [52].
- **Choreographed Saga** has no central coordinator. Instead, each service creates an event that will trigger the next subtransaction. More complex transactions can be harder to implement and understand when compared to *Orchestrated Saga*, but the availability is better due to not having a coordinator that is a potential point of failure.

Saga relaxes ACID in favor of Basically-Available, Soft-state, Eventually-Consistent (BASE) [61]. Given a proper application of Saga and BASE, some of the ACID guarantees can be supported to an extent:

- Relaxed atomicity is achieved by undoing the results of a failed transaction with compensating operations. For example, a compensating operation for creating a user would be deleting the user.
- Strong consistency is replaced by eventual consistency. A properly implemented Saga pattern guarantees that the system will eventually reach a consistent state.
- Isolation is not supported to any extent. Active transactions can see intermediate results of other active transactions.
- Durability can be supported if events are persisted. Replaying an event will execute the corresponding transaction once again. This can be useful when, for example, the effects of a transaction are lost.

Issues arising from favoring BASE over ACID are some of the major pain points for developers [48,45,66]. These include non-atomic message processing, feral ordering, and isolation anomalies. Still, despite the issues and the expressed importance of ACID by developers [45], Saga remains the most popular approach in MSA because the loose coupling of transaction participants is perceived as a good fit for MSA [48,45]. Despite claims within both white and grey literature that Saga scales better than 2PC, we have found only one research paper that directly compares the two approaches [26].

Saga only supports asynchronous, eventually consistent transactions, which is the exact opposite of what 2PC has to offer. It also has a large impact on software design by requiring the application to apply the EDA.

Transaction coordinators are a loose group of approaches for distributed transaction handling in MSA applications. Although this landscape is quite varied, all of the approaches either fall into synchronous, highly consistent and asynchronous, eventually consistent categories. Granola [13] and CloudTPS [70] use the classic locking mechanism of 2PC facilitated by a transaction coordinator to ensure high consistency across multiple heterogeneous data stores. Cherry Garcia [14] works on the same principle, but has no central transaction manager. Instead, the transaction state is embedded in the data store objects as a part of the metadata. This comes at the cost of Cherry Garcia being limited to only data stores that provide *Test-and-Set* operators, which enables atomic writes on an object in a single instruction. ReTSO [43] works by combining data-store meta

fields and centralized transaction manager to provide *SNAPSHOT_ISOLATION* consistency level. The transaction state is managed by the Timestamp Oracle component, which is a bottleneck and the single point of failure of the solution. Typhon [3] uses a separate layer for transactional metadata and instead relies on Vector Clocks [56] to avoid centralized timestamp generation, but only comes with causal consistency guarantees. GRIT [68] executes transactions optimistically, by first capturing read and write sets, before logically committing and physically materializing the changes in data stores. It comes with a large architectural requirement: not counting microservices and data stores, GRIT brings six additional components into the application. Furthermore, data stores must support multi-versioning and snapshot reads, further narrowing down the available options. All of the discussed approaches come with client libraries or large design limitations, like the aforementioned *Test-and-set* requirement of Cherry Garcia, or the specific technology stack imposed by Narayana. Ultimately, none fulfills the criteria of supporting highly consistent synchronous and asynchronous transactions with low impact on the codebase.

Federated databases (FDS) [65] and **Polystores** [31] tackle the data management issue by acting as a query execution layer on top of all data stores in a system. This way, a heterogeneous data store environment appears to be homogeneous to a service. A single language is used for querying and manipulating all the data in a system. A service working with data is unaware from which data store the data originates. This is a major upside of Polystores, which keeps services unaffected by the choice of data store technologies. However, there is an overhead added by underlying data migrations involved in query execution. The overhead is acceptable for OLAP workloads but is not favorable for short-lived transactions of OLTP workloads. Transactions on Polystores are asynchronous and eventually consistent, due to how data is replicated between data stores. The impact on the code is minimal as long as services are already using a querying language offered by the polystore. Polystores also bring a limitation to the choice of data stores, as only the ones supported by the polystore can be used.

Database transaction middlewares follow the principle of separation of concerns to bring transactional properties to heterogeneous data store environments. They are similar to FDS and Polystores from the perspective of layered data store approach. The general approach of this thread of work is to add a transactional layer to separate transaction execution from data storage, which is delegated to data stores. MIDDLE-R [59], Calvin [67], Bolt-On Consistency [5], Deuteronomy [50] follow this principle by adding a layer responsible for transaction execution, fault tolerance, and logical replication across different data stores, regardless of their transactional or replication capabilities. As a result, transactional properties and safety guarantees are unified across all data stores. This way strong consistency can be achieved for both synchronous and asynchronous transactions in MSA applications. SPTM takes this thread of work and applies it to microservices instead of data stores, but leaves the microservice source code intact.

Transaction middlewares for Function-as-a-Service (FaaS) are one of the more recent developments in the area of transaction management. Their primary objective is twofold: to act as a cache for objects used in ongoing transactions, and to provide a certain level of consistency guarantees. HydroCache [71] and FaaSTTC [54] are prime examples in this category that offer Transactional Casual Consistency level (TCC) [2], while also improving the overall performance of transactions in a FaaS context. While TCC is weaker than Snapshot Isolation, it is currently noted as being the highest con-

sistency level achievable without a consensus [4]. Transaction middlewares offer client libraries for reading and writing objects within a transactional context that need to be explicitly invoked in the client code. Strong consistency is supported for both synchronous and asynchronous transactions, but code and architecture need to be adapted to use the middleware.

In Table 1 we present an overview of related work compared by the characteristics defined at the beginning of this section. For this comparison, we consider the consistency level as high if it offers stronger guarantees than causal consistency [69]. TCC and *SNAPSHOT_ISOLATION* are examples of such levels. We show that the existing approaches do not cover the desired characteristics of synchronous and asynchronous support, high consistency, and low impact on the codebase and the software design. At best, some approaches, such as transactional FaaS middlewares, cover the first two characteristics. The latter is never considered, which is a gap that SPTM aims to address.

Table 1. Overview of related work

Approach	Sync/Async	Consistency	Code and design impact
2PC	Sync	High	- Client libraries - Data stores must all be transactional
Saga	Async	Eventual	- Must apply the EDA defensive coding due to BASE (e.g. message replay, atomic processing)
CloudTPS [70] Granola [13]	Sync	High	- Client libraries
Cherry Garcia [14]	Sync	High	- Client libraries data store <i>Test-and-set</i> operation
ReTSO [43] Typhon [3]	Both	High	- Client libraries
GRIT	Async	Eventual	- Architecture must be adapted to implement GRIT
FDS and Polystores	Async	Eventual	- Can only use data stores supported by the Polystore/FDS
Database transaction middlewares	Both	High	- Client libraries - Can only use data stores supported by the Polystore/FDS
Transaction middlewares for FaaS	Both	High	- Client libraries

3. Service Proxy Transaction Management — SPTM

In this section, we present the SPTM approach. First, we provide a high-level overview of the SPTM approach and provide a list of requirements that a system must fulfill in order to utilize SPTM. Then, we delve into more details on how SPTM assigns and identifies transactions and involved objects from messages. Afterward, we describe how SPTM can achieve fault tolerance and scalability. Next, we present a reference architecture for

the client-server messaging model. Finally, we highlight how SPTM differs from related work.

3.1. Overview of the SPTM Approach

Microservices provide a set of functions that can be invoked remotely, called endpoints. Similarly to regular functions, an endpoint defines a name, a set of parameters, and a return value. The name is usually given as a path or a Uniform Resource Locator (URL). The parameters and the results of an endpoint are sent via the network within messages. The format of these messages is defined in a *message schema*. Endpoints and rules on how to use them form an Application Programming Interface (API) of a service.

SPTM is an approach for transaction management with a focus on transparency to reduce the impact on microservice source code. It does so by fulfilling the following responsibilities:

- **R1 Message metadata retrieval:** SPTM retrieves the information on which messages can be involved in transactions and how data objects (DO) can be extracted and identified from them.
- **R2 Message interception:** Inbound and outbound messages are intercepted and their contents are inspected and modified.
- **R3 Transaction context and data object detection:** This includes transaction identifiers, transaction state transition commands, and DOs involved in the transaction.
- **R4 Transaction lifecycle management:** Transaction state is transitioned based on transaction commands or message success status.
- **R5 Data object version control:** DOs modified by a message are stored within a version control storage. This version control storage is then used to replace uncommitted DOs with their committed versions.
- **R6 Distributed state management:** The updated transaction and version control storage states are distributed to multiple nodes for scaling and fault tolerance.

The SPTM approach is designed to work with any type of messaging protocol, as long as it meets the following four criteria. First, messages must contain a metadata (header) section that can carry additional data, such as the HTTP header section. Second, messages must contain information identifying the name of the endpoint of a service. Third, messages must be able to carry a payload that can be parsed and inspected. Fourth, the messaging protocol allows for building a layered system, in which a participant cannot tell if it is directly communicating with another participant or not. Specifically, a proxy can be installed between any two participants.

The SPTM approach can be applied to both synchronous and asynchronous messaging patterns. Transactions in both synchronous and asynchronous usually begin with a synchronous message that confirms that the operation is accepted. This is sufficient for SPTM to build and detect the transactional context, as long as the followup messages belong to the same transaction.

SPTM is not an all-or-nothing choice and can be applied to a selection of endpoints in a system. Those endpoints need to satisfy the following three criteria. First, the endpoints are using a messaging protocol that meets the requirements of SPTM. Second, only the DOs identified from the messages involved in the endpoint are used in a transaction.

This includes all the messages generated during the endpoint execution, not just the request and the response exchanged in client-server communication. Third, the values in the metadata/header section generated and used by SPTM must not be modified after a transaction starts. They should also be propagated in all of the messages generated by an endpoint that are part of a single transaction.

3.2. Message Metadata Retrieval, Message Interception, and Data Object Detection

In this subsection, we delve into details about how SPTM deals with message metadata retrieval, message interception, and data object detection.

A DO is a key-value structure containing data about an entity handled by an application. Each key is associated with a field of the corresponding entity, with at least one field acting as an identifier (ID). The ID field can be used to uniquely identify the instance of the entity which is represented by the DO. DOs are carried within messages at the service communication level in a data serialization format. A DO representing a user in JavaScript Object Notation (JSON) format is presented in Listing 1. The user is uniquely identified by the *"id"* field and carries information about a person named John Doe.

Listing 1. An example of a user DO in JSON format

```
{
  "id": 123,
  "email": "johndoe@email.com",
  "firstName": "John",
  "lastName": "Doe"
}
```

SPTM works on the premise that messages exchanged at the service communication level contain enough information to determine what DOs are involved and the operation applied to them. This is particularly true for Representational State Transfer (REST) [29]. Consider a typical REST request *PUT /user/123*. One can conclude that this is an update on a user type, judging from the */user* prefix. The */123* suffix indicates that it is a user with identifier 123. *PUT* indicates that this is an update operation; a new version of the object will be created.

Operation type on a DO can be *CREATE*, *READ*, *UPDATE*, or *DELETE* (CRUD). Complex operations on multiple DOs commonly used in MSA can be defined as a composition of CRUD operations on individual DOs. To ensure atomicity, compensating operations can be defined as operations with the opposite effect. *CREATE* is compensated with *DELETE*, and vice-versa. *UPDATE* can be compensated with another *UPDATE* carrying the previous value of a DO. The limitation is that SPTM must be able to create a compensating operation out of the information available in the original operation. For example, a *CREATE* operation must contain an identifier that would be used by the compensating *DELETE* operation.

To our knowledge, there are no strict rules or standards on how a message schema is defined. Still, no matter the schema, messages can contain enough information to detect DOs and determine the operation type. A developer may, for example, choose the Remote Procedure Call style (RPC) over REST for user updates. RPC user update can be defined as *POST /updateUser*. The URL */updateUser* indicates that this is an update on the user object, while the message body carries a user DO that contains the identifier 123.

Since there are no certain assumptions that we can make about message schema at the service communication level, developers must provide the message metadata. SPTM does not have a prescribed way of gathering message metadata but has the requirement of low impact on the source code. We see several ways of achieving this. For example, analysis of a microservice's source code could reveal what endpoints are used and what DOs are involved in transactions. The same goes for API specifications, such as OpenAPI for REST [21], or Protocol Buffer definitions for gRPC [23]. A dedicated configuration file can also be provided to SPTM, which could be hand-written or generated from the source code. Finally, messages can be extended with metadata that can be evaluated on-the-fly. The on-the-fly message schema inspection is the most flexible as it can adapt to message schema changes, in contrast to other approaches presented here, which operate offline. On the other hand, the offline approaches do not bring as much runtime overhead, as most of the computation is already done before runtime.

SPTM has no information about the semantics of a model nor its constraints. Instead, it relies on microservice logic to check the correctness of the detected DOs. This is achieved by optimistically passing them to the microservice, which will determine the outcome of the operation. Subsequent read requests will have their responses replaced by either committed DOs, or DOs written by the transaction in which the request belongs. Allowing writes to finish lets SPTM avoid locking which is a commonly cited problem with 2PC.

In summary, DOs are detected within requests and responses based on a provided configuration. This configuration contains a list of involved services, their endpoints, what DOs they carry, and operation type. Operation types are the standard CRUD. Each endpoint definition also carries information about how to evaluate and identify DOs. For example, a *PUT /users/123* request applies an update based on the JSON message body to the user 123. Endpoints can optionally define a rollback (compensating) endpoint. For example, user creation can be compensated by user deletion.

3.3. Transaction Context, Transaction Lifecycle Management and Data Object Version Control

In this subsection, we delve into details about how SPTM deals with transaction lifecycle management and data object version control.

Since distributed transactions within MSA are spread across multiple remote calls, there is a need to correlate messages to a transaction. This information is carried within a *transaction context* via message headers. Specifically, transaction context is an identifier carried via *Begin-Txn*, *Commit-Txn*, *Abort-Txn*, and *Txn-Id* message headers. A Universally Unique Identifier (UUID) value is chosen because generating such an identifier does not require coordination. A transaction is created by an SPTM node in the *STARTED* state as soon as a *Begin-Txn* value is encountered for the first time, with the value used as the transaction ID. If a transaction with the detected ID already exists, the transaction is instead rejected and the client can choose to restart the transaction with a new *Txn-Id* value. This is the start transaction lifecycle as illustrated in Fig. 1. Subsequent messages carrying the same *Txn-Id* value belong to the same transaction. If any of the messages fail, the transaction transitions to the *FAILED* state and is aborted. Compensating messages are then sent where necessary and the transaction transitions into the *ROLLED_BACK* state if they all succeed. Otherwise, it transitions to the *ROLLBACK_FAIL* state. Transactions can

be committed or aborted by sending the transaction ID in the *Commit-Txn* or the *Abort-Txn* header, transitioning them to the *COMPLETED* or *FAILED* states, respectively.

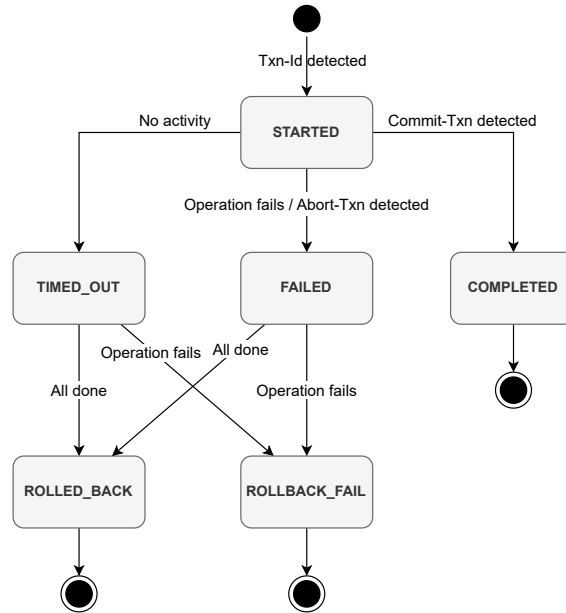


Fig. 1. SPTM transaction lifecycle

SPTM maintains a version chain with multiple versions of a single DO. When a transaction modifies a DO, a new version of the DO is added to the version chain. Transactions can access adequate DO versions using transaction timestamps, which can be physical or logical. Using physical timestamps over logical timestamps could be considered more practical, but runs the risk of clock skew if timestamp generation is not centralized [47]. A timestamp is generated when a transaction starts, which is then used for subsequent operations, regardless of the physical time of the operation. Transactions can only access committed DO versions with starting time before and ending time after the timestamp, or written by themselves.

Conflict detection is initiated on a transaction commit. SPTM goes through versions of DOs involved in a transaction and looks for write-write dependencies in the following manner. Consider a scenario with a transaction $T1$, that writes a new version $v1$ on DOI at time $t1$. A write-write dependency is detected for transaction $T1$ if there is a transaction $T2$ at time $t2$ that: (i) is in the *COMMITTED* state, (ii) has written a new version $v2$ on DOI , and (iii) $t2 > t1$. Read-write and write-read dependencies are necessary for *SERIALIZABLE* isolation level [60], but would require a consensus for reads as well, which would adversely affect performance. Therefore the maximum supported isolation level offered by SPTM is *SNAPSHOT ISOLATION*.

Conflict resolution is done in the first-writer-wins or last-writer-wins approach. Normally, the last-writer-wins approach carries the risk of messages being ordered differently

than what SPTM had observed, potentially violating the happens-before relation between transactions. Consider a scenario in which SPTM receives a message from $T1$ writing a version $v1$ of DO at $t1$, and $T2$ writing a version $v2$ at $t2$. SPTM observes that $t1 < t2$, but once messages are sent out to the microservice, they can be processed in any order. The risk is increased with the number of components involved in the processing, which is commonly at least two: microservice and its data store. Furthermore, even if the order of processing on all of the involved components stays identical, the responses could arrive in a changed order. Consider that t_{w1} and t_{w2} are the times of response for write operations and t_{db1} and t_{db2} are data store write times of $T1$ and $T2$ respectively. If $t_{db1} < t_{db2}$, but $t_{w2} < t_{w1}$, SPTM would conclude that $t2 < t1$ and abort $T2$. This is not a problem as long as all read and write operations are served through SPTM, as it possesses a consistent snapshot in which $T1$ is committed and $T2$ is aborted. First-writer-wins approach might be a better choice if retrying a transaction is less expensive than compensating it, for example when compensations are needed for aborted transactions. However, first-writer-wins effectively behaves pessimistically: all newly started transactions modifying a DO that is modified by an active transaction will be aborted.

Traditional database transactions are directly tied to database connections: when a database connection is closed, the transaction ends. With SPTM, transactions can span many independent messages. Services can fail while processing messages, leading to abandoned transactions. Abandoned transactions take up resources indefinitely and can cause all future transactions to be aborted due to conflicts, should the SPTM implementation use first-writer-win conflict resolution. Transactions that have not seen activity for an extended period will transition into the *TIMED.OUT* state and are treated identically to *FAILED* transactions.

Fig. 2 gives an overview of the concurrency control scheme described by SPTM. For simplicity, we use integers both for transaction identifiers and timestamps. The blue value is the latest committed value, while the green value is the value written by the ongoing transaction started by the green writer. Both readers access the value at timestamp $T=6$, which falls within the timespan of both versions. The green reader can see uncommitted values written by the green writer due to them being in the same transaction. The blue reader cannot see the uncommitted value because it is a part of another transaction.

3.4. Distributed State Management

In this subsection, we delve into details about how SPTM can deal with distributed state management.

SPTM is able to operate on multiple nodes by replicating transaction state changes and DO version chain updates. There are several potential points at which updates can be sent to the cluster:

1. **Option 1 — After each message:** SPTM sends an update containing the new transaction state and newly extracted DOs whenever a new message is processed.
2. **Option 2 — On transaction state change:** SPTM sends the new transaction state and DO deltas only when a transaction is transitioned to a new state.
3. **Option 3 — Periodically:** SPTM sends transaction state and DO deltas at fixed time periods, when a certain number of operations has been executed, or using a combination of both.

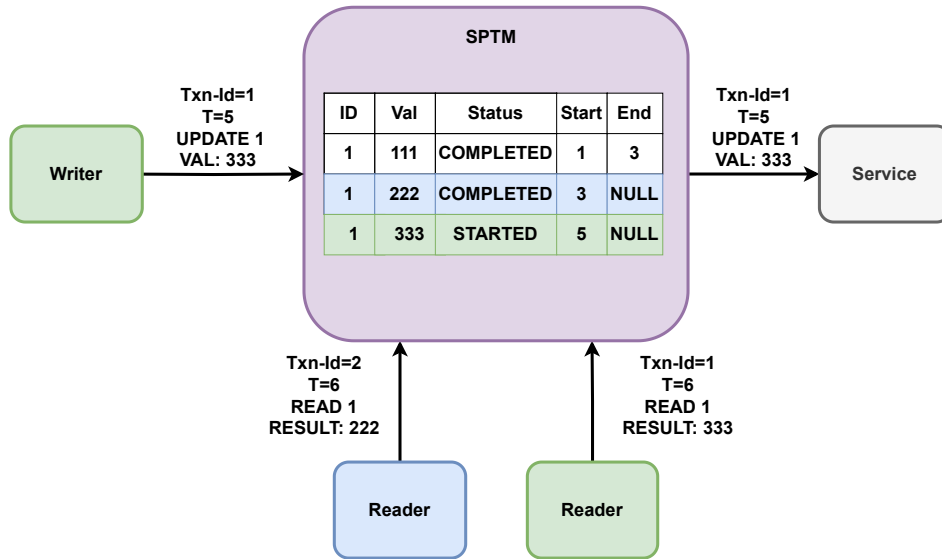


Fig. 2. Concurrency control overview of SPTM

Options should be chosen carefully based on architecture and workload profiles. For example, Option 1 offers the lowest replication lag, but also incurs the highest overhead because updates are sent out on each update. It is suitable for architectures in which multiple nodes can work on a single transaction with high consistency. On the other hand, shared-nothing architectures would benefit more from Option 2 and Option 3. The key design choice here is whether to partition the system on DOs or transactions. Partitioning on DOs is desirable for traditional OLTP workloads in which there are groups of interdependent DOs that are frequently processed together [44]. Such DOs can be collocated on a single node for highly efficient in-memory operations. Distributed state change is necessary only when a transaction operates on multiple unrelated DOs, or to replicate state for fault tolerance. On the other hand, partitioning on transactions could be more efficient when processing multiple unrelated DOs, but has a risk of conflicting writes on DOs.

3.5. Reference Architecture

In this subsection, we present a reference architecture that can be used to implement an SPTM transaction manager. In this particular case, we focus on the client-server communication model. Fig. 3 illustrates the reference architecture and its six components: one for each of the responsibilities defined in Section 3.1.

Metadata retriever is the component that fulfills R1. It reads message metadata from a configuration file, which is either generated from the API specification of a microservice or manually created by its developers. Metadata information is then passed on startup to an SPTM node to be used by other components. We use denoted lines in Fig. 3 for this activity to emphasize that it does not happen during runtime.

Message interceptor is the component that fulfills R2. It is a proxy that can interpret, load, and modify messages of the chosen protocol. Once it detects a request message

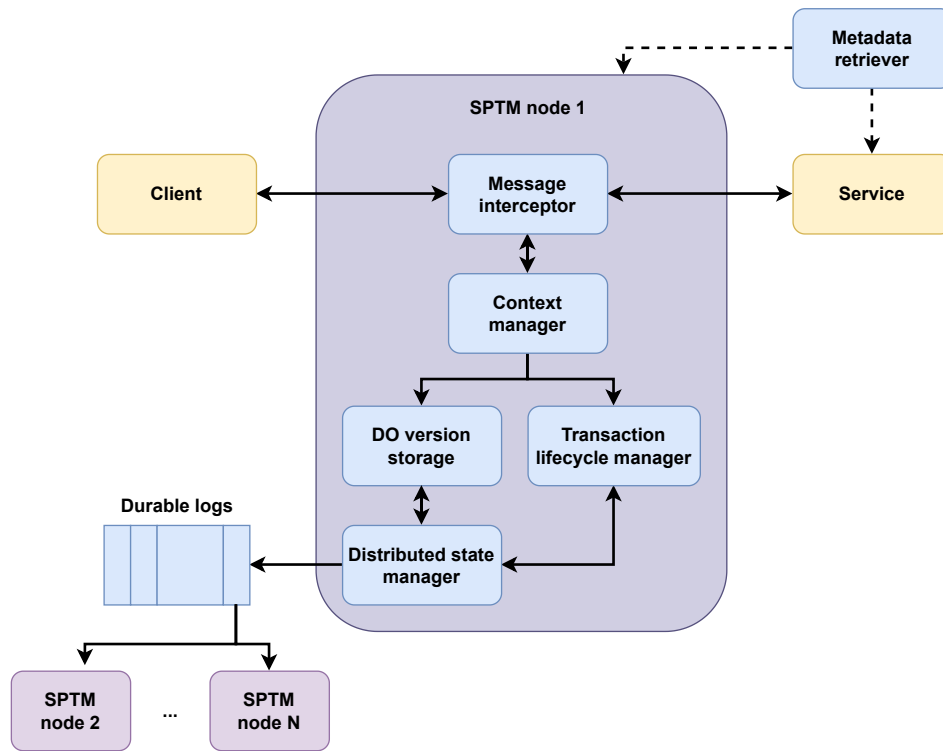


Fig. 3. SPTM reference architecture

matching a rule in the configuration file, it loads the payload and additional data, such as headers, into memory to be processed based on the transaction context. Once the processing is finished, a new connection to the intended microservice is created and the content modified by other components is sent through it. After receiving the response message, the payload, headers, and status are sent to the context manager. If the transaction fails, the response message is modified to reflect transaction failure, usually by setting the appropriate status code. Otherwise, the response message is loaded and processed identically to the request message. The only exception is that the modified content is sent back to the connection that carried the request message instead of creating a new one.

Context manager is the component that fulfills R3. It builds a transaction context, which consists of transaction information and DOs involved in the message. Transaction information carries the transaction timestamp, which the context manager uses to fetch DO versions available to the transaction. These DO versions are then used by the message interceptor to replace the original DOs found in the message payload.

Transaction lifecycle manager is the component that fulfills R4 and is responsible for keeping track of transaction information. This includes their status, timestamp, and dependencies. It is also responsible for transitioning the transaction state and triggering the corresponding actions, such as sending out compensating messages when specified.

DO version storage is the component that fulfills R5 and is responsible for managing DO versions and providing DO versions for a given timestamp. It is also responsible for detecting conflicting DO operations done by different transactions.

Distributed state manager is the component that fulfills R6. It manages the distributed state through the use of durable logs. A durable log contains basic information about a state update and is first stored on a local disk to ensure durability. At first, the log is uncommitted and its effects are not applied. It is then distributed across the network to other SPTM nodes: nodes two through N in Fig. 3. These nodes persist the durable log on their local disks and apply the changes, before sending back a confirmation message. Once the number of confirmations reaches a certain threshold, e.g., the majority of nodes have sent a confirmation, the durable log is committed and its effects are applied.

3.6. Differences to Related Work

In this subsection, we present the rationale behind SPTM and how it differs from existing approaches.

There are two key differences between the existing approaches and the SPTM approach. First, SPTM strives to be non-invasive, with little-to-no impact on the code. This lifts the burden of distributed transactions from developers, allowing them to focus on application logic. Second, SPTM intercepts, evaluates, and modifies messages at the service communication level. Other approaches operate at the database communication layer: as a proxy to a data store, by embedding metadata in the data store, or as a standalone component that the service must use.

We chose SPTM to operate at the service communication level because REST style with HTTP appears to be the *de facto* standard in the industry. In Table 2 we show the technology choices of eight open-source MSA projects. The table, alongside industry studies [9], reveals that HTTP/REST is the only constant across all services, unlike data stores that are more varied. Relational databases are most common but are too different even among themselves. Supporting all of them with a single solution is not a simple task: it remained an unresolved challenge of FDS [11].

The results of industry studies cannot be extended to the entire industry due to sample size. However, we believe they provide a good case that there is much less variance in service level protocols than in data stores. Operating at the service communication level allows SPTM to cover more ground with a single implementation. The SPTM approach is applicable to any protocol and style beyond HTTP/REST, such as gRPC [20] or GraphQL [19]. Furthermore, it allows SPTM to support both synchronous and asynchronous messaging, as long as the chosen messaging protocol satisfies the requirements of the SPTM, as defined in Section 3.1.

4. Implementation

As a part of our research, we implemented a transaction manager called *fed-agent* that follows the reference architecture defined in Section 3.5. It is written in the Go programming language [18], with a focus on the REST architecture and JSON DOs.

Message metadata is provided in a configuration file and is built using the same building blocks used to define a microservice endpoint. These are the URL, the message body,

Table 2. Overview of technologies in open source MSA projects

Project	Languages	Messaging	Data stores
eShopOnContainers [32]	C#, .NET	HTTP REST, RabbitMQ	SQL Server, Redis
ESPM [33]	Java, Spring	HTTP REST, RabbitMQ	MySQL, Redis
LakesideMutual [35]	Java, Spring	HTTP REST, RabbitMQ	H2
FTGO [34]	Java, Spring	HTTP REST, Kafka	MySQL, DynamoDB
Vert.x Blueprint [39]	Java, Spring	HTTP REST, Kafka	MySQL, MongoDB, Redis
Sentilo [36]	Java, Spring	HTTP REST, Kafka	MongoDB, Elasticsearch, Redis
Spring Pet Clinic [38]	Java, Spring	HTTP REST	HSQLDB, MySQL
Sock Shop [37]	Java, Spring, Go	HTTP REST, RabbitMQ	MySQL, MongoDB

the headers, and the method. In Listing 2 we show a configuration example for a single HTTP/REST endpoint. *PUT /user/{id}* request is an *UPDATE* operation on a user identified from the path parameter *id*. The *request* field describes the content of the request. It is a JSON object, as described by the *content_type* field. It contains a single user DO, that can be identified by reading the *id* property of the JSON object within the message body. The configuration also describes a compensating operation in the *rollback* field. The *target* field contains the name of the endpoint that will undo the effects of this operation. It is the same endpoint, but the *data* field has a *data_source* subfield specifying how to form a compensating operation. In this instance, the last committed version of the DO should be used in the request body of the compensating operation.

Not all messages need to be intercepted by *fed-agent* though and it should be positioned within the architecture in a manner that only adds overhead where necessary. More specifically, only messages involved in OLTP workloads should be considered. When an HTTP request is received, *fed-agent* first checks if the URL is specified in the configuration. If not, the request is passed through to the microservice and is ignored by *fed-agent*. Otherwise, *fed-agent* copies the payload and deserializes its JSON payload for inspection before passing the unmodified request to the microservice. After receiving the response, *fed-agent* does a series of operations depending on the operation type. In the case of writes, *fed-agent* first checks the response code. If the code is not in the success range, i.e. *2XX*, the transaction state is set to *FAILED*, and the update is sent via consensus. Otherwise, DOs are extracted from the request payload based on the configuration rules and added to the DO version store. The new versions are then sent out via consensus and the state is reconstructed by the followers. For reads, *fed-agent* deserializes the JSON payload and extracts all the DOs from it. Then it finds the last committed version of each DO and replaces their occurrences in the payload before sending the modified response payload to the client.

Listing 2. *Fed-agent* configuration example for a user update endpoint

```

{
  name = "update-user-profile"
  idempotent = true
  method = "PUT"
  path = "/user/{id}"
  type = "UPDATE"
  request {
    content_type = "json"
    entities {
      user {
        id_source = "body"
        id_path = "id"
      }
    }
  }
  response {
  }
  rollback {
    target = "update-user-profile"
    data {
      content_type = "json"
      entities {
        user {
          data_source = "version"
          data_target = "body"
        }
      }
    }
  }
}

```

To provide a high level of isolation and consistency, *fed-agent* needs to ensure that intermediate results of active transactions are not visible to other transactions. For this purpose, a concurrency control scheme needs to be in place. We chose Multi-version concurrency control (MVCC) [7] using timestamp ordering (MVTO) [72] as the concurrency control scheme in *fed-agent*. There are also two other MVCC variants: two-phase locking (MV2PL) and optimistic (MVOCC). We find these less suitable due to the following reasons. MV2PL acquires locks on DOs, which can lead to deadlocks and connection exhaustion. MVOCC works by holding on to changes until the end of the transaction, at which point all of the changes are sent to storage, or in this case, services. This is not desirable for two reasons. First, *fed-agent* has no information about data constraints and validation: they are delegated to the services. Therefore, *fed-agent* cannot determine the validity of DOs in a message until it gets a response from a service. Second, applying all changes at the end of a transaction can generate a burst of messages in a short period. The pressure created by a large burst of messages can disrupt microservices, leading to failed messages. Failed requests can trigger compensating messages, which amplifies the issue. However, as an SPTM implementation, *fed-agent* keeps a consistent snapshot of DOs and compensating operations are not necessary if all reads and writes go through *fed-agent*. We leave this as an option to allow for eventual consistency of DOs, in case they need to be accessed directly in data stores.

Fed-agent periodically runs a background routine that deletes data from finished transactions and progresses active transactions toward their finished state. A transaction is in a finished state if its status is *COMPLETED*, *ROLLED_BACK*, or *ROLLBACK_FAIL*. *Fed-*

agent first goes through finished transactions and deletes their DO versions from the DO version store. The only exceptions are DO versions from *COMMITTED* transactions that are the last committed version for their respective DOs. Then, *fed-agent* tries to progress currently active transactions. A transaction is moved from *STARTED* to *TIMED_OUT* state if there have been no operations within a configured period. Compensating operations for *TIMED_OUT* and *FAILED* transactions are attempted for a configured number of retries or until a configured period is passed before transitioning to *ROLLED_BACK* or *ROLLBACK_FAIL*.

With *fed-agent*, we chose Option 1 from Section 3.4 for distributed state management. *Fed-agent* uses the Raft [58] consensus protocol for transaction tracking. A *fed-agent* cluster has one leader and multiple followers that can serve reads. Updates to transaction statuses and the version chain are accepted by the leader and replicated to followers on each intercepted message. Having more followers increases availability, fault tolerance, and read parallelism at the cost of write performance. Should the leader become unavailable, one of the followers can become the new leader. The new leader then triggers the recovery process for in-flight messages: messages that were detected and passed to a service, but a response was not registered. Since no response is received, the transaction state cannot be determined and it must be either replayed or aborted. If a message is marked as idempotent in the configuration, it can be replayed safely. Otherwise, the corresponding transaction must be aborted to maintain consistency.

Transaction updates are made durable by writing a log to persistent storage. A log contains information about a non-read transaction operation, namely its type, timestamp, and data. After accepting a non-read operation, the leader writes a log to the local disk and sends it over the network to followers. Upon receiving the log, the followers also write the log to disk and send a confirmation back to the leader. Once the majority of nodes have confirmed a log write, the transaction state is successfully updated. The more nodes there are in a cluster, the longer this process takes. However, having a larger cluster increases availability, as a cluster will continue to operate as long as the majority of nodes are online. Larger clusters also offer improved read parallelism, due to the higher number of followers that can serve reads. This trade-off between write performance and availability plus read parallelism can be tuned by adjusting the size of the cluster.

In Fig. 4, we depict a scenario in which the results of a write operation served by the leader are replicated to followers to be provided to readers. Dashed lines are used to denote replication via consensus protocol, while solid lines are used to denote direct communication from transaction participants. After the green writer creates a new DO version, it is replicated to both followers. The green reader can then read the new value without accessing the leader.

Fed-agent has two distinct modes of operations: single-node and multi-node. In multi-node mode, *fed-agent* creates a Raft consensus group with a single leader and multiple followers. All write operations are handled by the leader since they can affect *fed-agent*'s state by modifying a transaction's status and the DO version store. Physical timestamps are used to order these updates and are only generated by the leader, which eliminates the risk of clock skew at the expense of write throughput. Operations are sent out to followers, which then execute them in the ascending timestamp order to construct the global state. In contrast, read operations do not generate any further network communication as they do

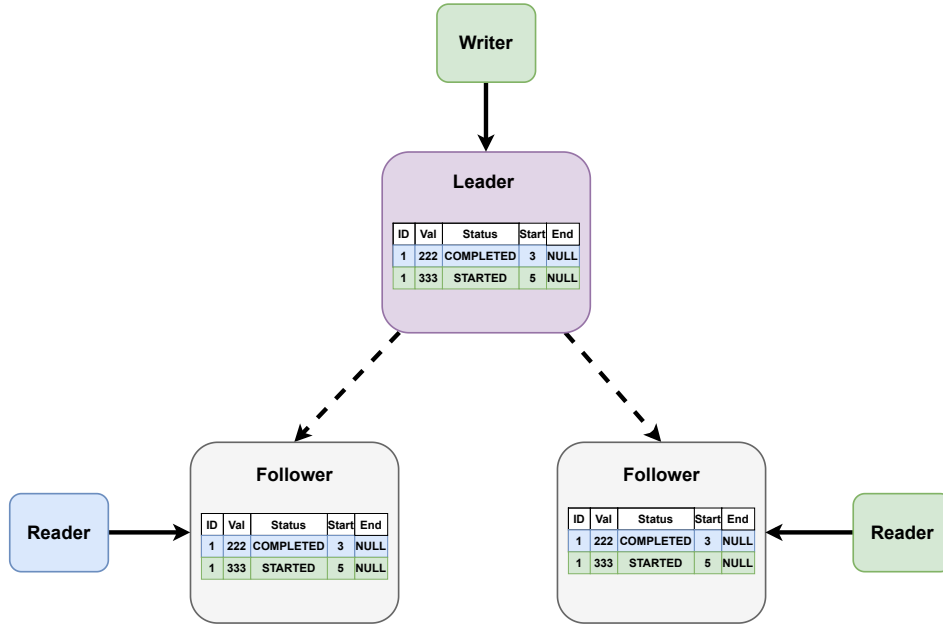


Fig. 4. A *fed-agent* cluster consisting of a leader serving writes and followers serving reads

not modify the state of *fed-agent*. Single-node mode avoids all consensus-related modules and directly writes updates to the local storage for persistence.

Fed-agent mostly operates in-memory to minimize its overhead. We believe that the in-memory storage architecture is well-suited for this case, as the memory footprint of active transaction data is usually small enough to fit even into the memory of programmable network switches [51]. DOs from active transactions are stored in a DO version store in the memory of the *fed-agent* process, which offers *SNAPSHOT* isolation level [1]. The minimum information necessary for crash recovery is stored within logs on the persistent storage on each transaction operation. These logs can be replayed to restore the pre-crash state of *fed-agent*. Periodic snapshots of the state are also stored to reduce the number of logs that need to be replayed during recovery.

5. Evaluation

In this section, we evaluate SPTM by comparing *fed-agent* to 2PC and Saga as main representatives of their categories, due to how ubiquitous they are in MSA applications. For both comparisons, we used mostly identical setups, with only a message queue component added in for Saga. Transactions used in the benchmarks are implemented to result in the same state in both synchronous and asynchronous versions, used by 2PC and Saga comparisons respectively.

5.1. Setup

We ran benchmarks on an e-commerce system for an online video game. In this system, a user can buy appearances for their in-game characters, called *skins*. There are four services involved in buying a skin: *store-service*, *payment-service*, *game-service*, and *gateway-service*. The responsibilities of services are as follows: *store-service* keeps track of user credits and skin prices, *payment-service* manages payments, *game-service* stores a list of skins available to a user, and *gateway-service* coordinates transactions made by a user. Each service is a REST service implemented in Go that uses PostgreSQL 13 [22] as the underlying database in a database-per-service fashion. Only the standard Go libraries and low-level database drivers are used for the implementation.

We define *BuySkin* transaction offered by *gateway-service*. The workflow is shown in Fig. 5:

1. *GetUserBuyInfo*: Sends a request to the *store-service* to check if the user has the credit to buy the skin.
2. *MakePayment*: Create a payment with a value equal to the skin's price and send it to *payment-service*.
3. *AddUserSkin*: Make the skin available to the user by updating the user profile in *game-service*.
4. *UpdateUserCredit*: Redact price from user credits and send an update to *store-service*.

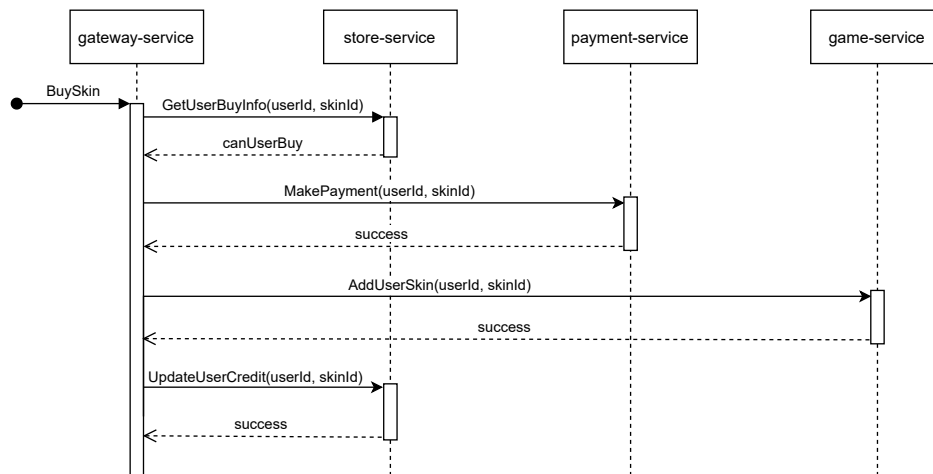


Fig. 5. Sequence diagram of *BuySkin* transaction

The setup was deployed on the Amazon EC2 service with each service, database, and client deployed on a separate *t3.2xlarge* instance (8vCPU, 32GB RAM) [15]. The isolation level of PostgreSQL transactions was set to *REPEATABLE READ* so that 2PC matches the isolation level of *fed-agent*.

Benchmarks are executed by starting an HTTP client calling the *BuySkin* endpoint with a random skin ID. Since transactions can abort due to a conflict, the client retries the

transaction until it succeeds with a two-minute timeout. Time until transaction success is measured and referred to as execution time. We also track the number of aborted transactions to help understand the execution time better. We compare execution time for 2PC, Saga, and *fed-agent* on the following dimensions:

- **Concurrency:** number of parallel clients, i.e., threads calling the *BuySkin* endpoint.
- **Contention:** number of concurrent clients attempting to modify the same DO, controlled via distribution. Distribution defines the probability for a DO to be accessed by a transaction. It can be uniform, hotspot, or Zipfian. Uniform distribution chooses a DO uniformly at random, while Zipfian chooses a DO according to the Zipfian distribution, resulting in a small number of popular DOs. Hotspot distribution defines two parameters: the fraction of operations accessing “hot” DOs and the size of the “hot” dataset. Uniform and Zipfian distributions are the most common distributions in web services [12], while hotspot gives us better control over contention scaling.

We formulate the following scenarios for benchmarks:

- **Scenario 1 — Uniform distribution:** the number of parallel clients is scaled from 0 to 200 using a uniform distribution. The purpose is to examine the behavior of systems under high concurrency and low-medium contention.
- **Scenario 2 — Zipfian / extreme contention:** The number of parallel clients is scaled from 0 to 60 using Zipfian distribution. The purpose is to examine the behavior of the systems under extreme contention.
- **Scenario 3 — “Hot” dataset:** The number of parallel clients is 100 using hotspot distribution. We define a “hot” dataset of DOs as 2.5% to 10% of the total dataset, which is accessed by 20% of all operations. The purpose is to examine the impact of scaling contention level on the behavior of the systems.

Each run of each scenario executes 30,000 transactions on the dataset of 1,000 skins and one user. Since a user-skin reference is considered a standalone object, having exactly one user is sufficient to test concurrency and contention. It also allows us to better control those two parameters by keeping the user fixed and changing the distribution of skins.

5.2. Comparison with 2PC

In this subsection, we compare *fed-agent* with a 2PC implementation by comparing the transaction execution times of both. A 2PC implementation is close to the *fed-agent*: a distributed transaction is a sequence of HTTP requests. This makes direct comparison with 2PC implementation rather straightforward.

We only compare a single-node *fed-agent* cluster to 2PC for one main reason. During our testing, we noticed that having multiple 2PC coordinators does not improve transaction execution time. This is because the main work is done by databases, not the coordinators. The coordinators only facilitate the transactions by opening and maintaining connections to databases that have a finite connection pool. Multiple coordinators enable the system to open connections faster, practically letting the system exhaust the connection pool earlier. Since the goal of this benchmark is to measure transaction execution time, developing a highly complex, multi-node 2PC implementation will not meaningfully contribute to the comparison.

This comparison employs two separate *BuySkin* versions in *gateway-service*: one using 2PC (*BuySkin2PC*) and one using *fed-agent* (*BuySkinFedAgent*). Both versions bring minor tweaks to the original *BuySkin* transaction. For *BuySkin2PC*, *gateway-service* generates and sends a *Txn-Id* header containing a transaction ID with each request. Services correlate this value to a local database transaction that is active until the global transaction finishes. For *BuySkinFedAgent*, *gateway-service* generates and sends a *Txn-Id* header containing a transaction ID with each request. *Fed-agent* takes care of the transaction management with no additional changes to services.

It should be noted that each run had to be split into 15 iterations of 2,000 transactions due to 2PC implementation blocking indefinitely after a certain time under high loads. *Fed-agent* was more resilient in this regard and would continue working, but would block for almost a minute before continuing due to high IO activity on the *fed-agent* node. We attribute this to benchmarks hitting the Amazon AWS I/O throughput limit for the instance types used [15].

Fig. 6a and Fig. 6b contain benchmark results for low contention (under 20% aborts) and increasing concurrency. *Fed-agent* is slightly slower than 2PC when the number of concurrent users is low. *Fed-agent*'s overhead is relatively constant compared to execution time, making it more noticeable at low loads. As the load increases, *fed-agent* execution becomes almost twice as fast as that of 2PC. *Fed-agent* exhibits a 20% aborted transaction rate which, while twice as high as that of 2PC, is still low enough that it does not lead to performance degradation.

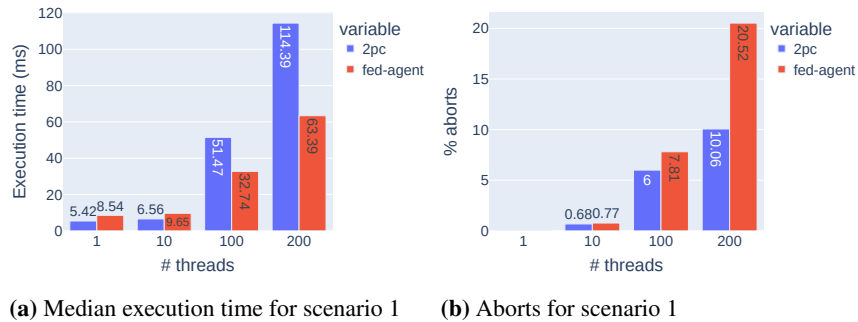


Fig. 6. The effects of the number of concurrent clients for scenario 1 (uniform distribution)

Fig. 7a and Fig. 7b contain benchmark results for extreme contention and moderate concurrency (scenario 2). The results show that 2PC and *fed-agent* behave comparably for abort rates up to 90%. Beyond that point, *fed-agent* starts to fall behind. We attribute this primarily to *fed-agent*'s usage of MVOCC, which is inherently worse for high abort rates than MV2PL used by the combination of PostgreSQL and 2PC.

We have observed that roughly 10% of 2PC transactions were dropped and never completed in the timeout period for 40 and 60 thread benchmarks. That likely happened because services maintain open database connections throughout the execution of a trans-

action, exhausting the I/O resources of the EC2 instance [15]. *Fed-agent* did not have this problem because database connections are short-lived in this case. We believe that this is an indication of the bottleneck of 2PC that prevents it from scaling.

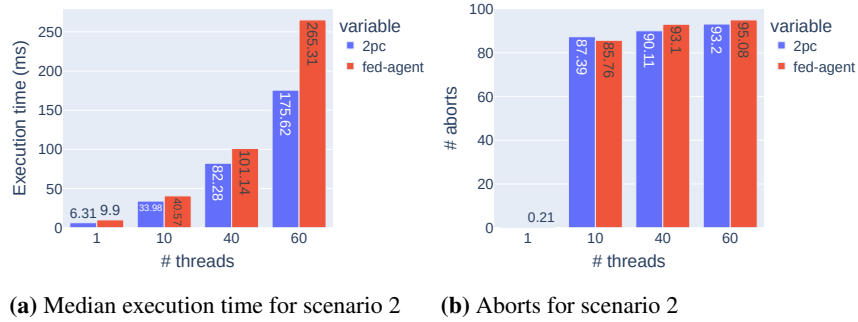


Fig. 7. The effects of the number of concurrent clients for scenario 2 (Zipfian / extreme contention)

Fig. 8a and Fig. 8b show the effects of contention on execution time (scenario 3). Contention is scaled by keeping the number of concurrent users at 100, while scaling the “hot” dataset from 20% to 2.5% of the entire dataset. Each operation has a 20% chance of accessing a DO from the “hot” dataset in all runs. We see that *fed-agent* performs slightly better than 2PC until roughly 60% abort rate. *Fed-agent* starts to quickly fall behind beyond 70% abort rate for 100 concurrent users.

Scenario 2 and scenario 3 benchmarks reveal that rollbacks have the highest impact on *Fed-agent*’s performance. Fig. 9a and Fig 9b show the total number of transaction attempts during the execution of scenario 2 and scenario 3 respectively. Despite a similar abort percentage, the total number of requests is disproportionately higher for *fed-agent*. For example, for 60 concurrent users using Zipfian distribution, for a 2% difference in abort rate, there is a 30% difference in total attempts. We suspect this discrepancy was caused by two things. The first is the first-write-wins conflict resolution of the *fed-agent*: once a transaction has written a new value for a DO, all transactions attempting to update the same DO are aborted. The second is *fed-agent*’s eager conflict detection: transaction conflict detection is done on each transaction status update, not just before committing. This is an implementational choice not imposed by the SPTM approach. Both properties can lead to a high number of short-lived aborted transactions, which is potentially detrimental to performance for high abort rates. Future research will focus on lowering the number of aborts or sending fewer compensating messages whenever possible.

In conclusion, *fed-agent* performs better than 2PC in high concurrency, low-medium contention scenarios while offering similar isolation and consistency levels. 2PC outperforms *fed-agent* in scenarios with high abort rates. It should be noted that *fed-agent* rollback mechanism always sends compensating operations when a transaction fails. 2PC is not safe from this either: if a database transaction fails to commit in the commit phase, other database transactions within the distributed transaction need to be compensated. The

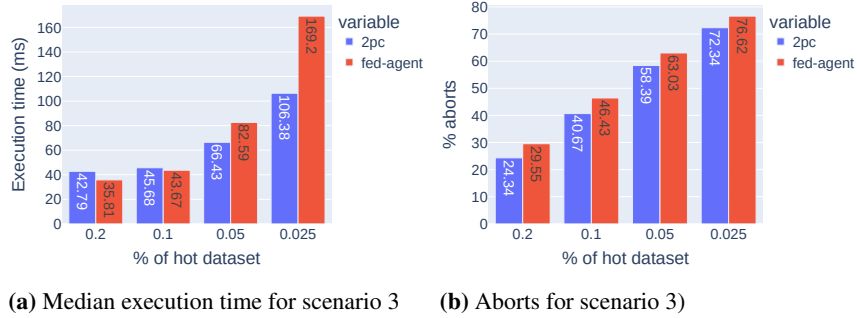


Fig. 8. The effects of contention for scenario 3 (“hot” dataset)

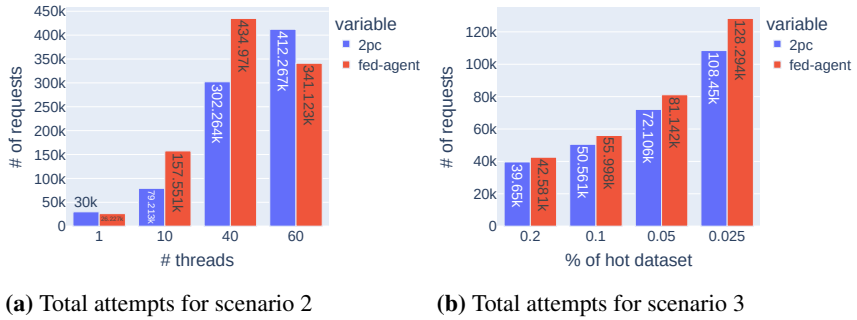


Fig. 9. Total number of requests scenario 2 (Zipfian / extreme contention) and scenario 3 (“hot” dataset)

2PC implementation used in benchmarks does not support this. The absence of rollback messages could have a noticeable impact on the performance for high contention and bring 2PC execution time closer to that of *fed-agent*. The biggest caveat to using a 2PC is the fact that it is only applicable if all underlying databases are transactional. *Fed-agent* does not have this requirement. Furthermore, 2PC was observed to stop accepting transactions after hitting a certain amount of requests, most likely caused by the connection pool exhaustion and locking. The benchmarks on *fed-agent* show that an implementation of SPTM can offer high consistency while being scalable and non-intrusive to the application code.

5.3. Comparison with Saga

In this subsection, we compare *fed-agent* with a Saga implementation by comparing the total execution time of all transactions, rather than individual execution times. We take both finished and compensated transactions into account.

The Saga implementation uses RabbitMQ [24] to pass transaction messages between participants. We only compare a single-node *fed-agent* cluster to a single-node RabbitMQ

cluster with default configuration and durable message queues. The RabbitMQ instance is hosted on a separate machine. RabbitMQ clients utilize a connection-per-thread approach instead of a channel-per-thread approach to maximize throughput. The number of open connections is equal to the number of client threads used in the benchmark.

This comparison employs two separate *BuySkin* versions in *gateway-service*: one for Saga (*BuySkinSaga*) and one for *fed-agent* (*BuySkinFedAgent*). *BuySkinFedAgent* is identical to the one used in comparison with 2PC. *BuySkinSaga* is a choreographed Saga implementation of *BuySkin* that uses RabbitMQ messages. We choose the choreographed approach to avoid additional message passing between participants and the coordinator that would negatively affect the performance. Each subtransaction of *BuySkinSaga* has a compensatory transaction that undoes its effects, while *BuySkinFedAgent* does not need them. Each subtransaction also sends an acknowledgment back to the RabbitMQ cluster once a message is successfully processed, or a negative acknowledgment if it fails. Both *BuySkin* variants are started by the *gateway-service*, with *BuySkinFedAgent* synchronously returning a response once the transaction finishes and *BuySkinSaga* being asynchronous, returning a success when a transaction is accepted. In both cases, a transaction is started by an HTTP request from the test client. We also modify benchmarks so that Saga executions can detect conflicts by adding a unique constraint on the relationship between user and weapon skins.

Fig. 10a contains benchmark results for low contention (under 20% aborts) and increasing concurrency, and Fig. 10b contains benchmark results for extreme contention and moderate concurrency (scenario 2). Both benchmarks reveal that Saga vastly outperforms *fed-agent* when the number of threads is less than 20 due to the disparity of synchronous and asynchronous message processing in the test setup. The test client works by sending HTTP requests sequentially in the specified number of threads. Since *fed-agent* transactions appear to be synchronous to the clients, the system's resources will remain underutilized as transactions are started one by one in a low number of threads. On the other hand, starting a Saga transaction is much faster because the *gateway-service* only needs to send a RabbitMQ message and return a positive response. As a result, the test client starts all of its transactions much faster in the lower thread range, allowing the system to better utilize its resources. This difference is gone as soon as the system becomes saturated with more concurrent client threads (>20). *Fed-agent* has identical execution times across all of the scenarios as Saga when the number of threads is over 20, while also offering higher consistency.

Fig. 10c show the effects of a small "hot" dataset on execution time (scenario 3). *Fed-agent* has no need for compensations, leading to a 50–60% faster execution than Saga. Saga degraded in performance in this scenario but not in scenario 2 because scenario 2 did not reach the concurrency level needed to cause a high number of compensations.

In conclusion, our benchmarks show that Saga vastly outperforms *fed-agent* in scenarios under 20 concurrent users. This happens because transactions are started much faster asynchronously by the test client when the system is not saturated. Both perform equally in low-to-moderate contention scenarios, with *fed-agent* offering high-consistency and ACID properties. In high-contention scenarios, *fed-agent* outperforms Saga by a wide margin, due to Saga needing to compensate failed transactions, whereas *fed-agent* does not. We also report that the implementation of a choreographed Saga is more challenging

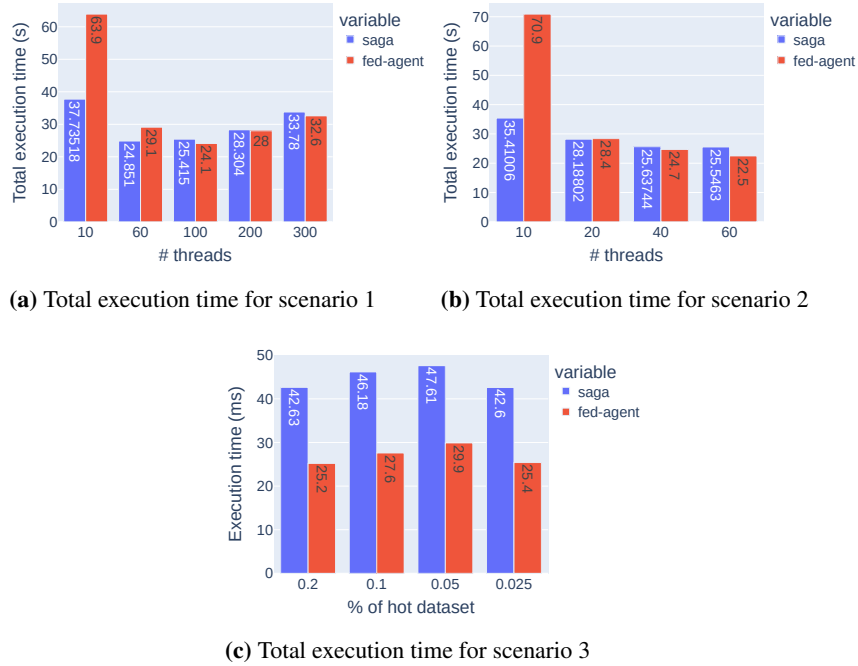


Fig. 10. Execution times of *fed-agent* and Saga across all scenarios

than that of *fed-agent* because the developer needs to take care of message acknowledgment, compensations, retries, and tracking of decentralized transaction execution.

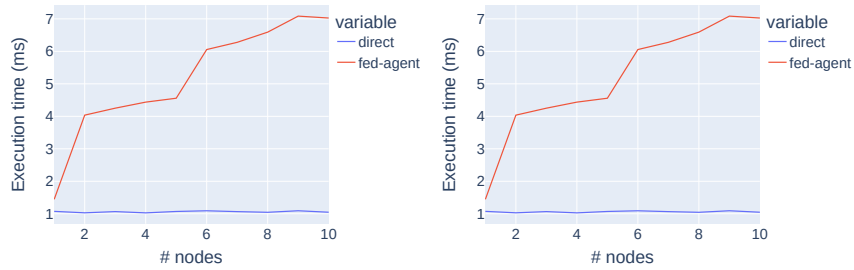
5.4. Overhead Characteristics

In this subsection, we measure the overhead of *fed-agent* as the difference between the execution time of direct microservice calls and calls via *fed-agent*. We consider the impact of cluster size and payload size on read and write overheads. We run all benchmarks in a single thread to better isolate the overhead from other factors, such as concurrency and contention.

In Fig. 11 we show that a single-node *fed-agent* cluster adds 0.5ms to reads and 2ms to writes. Moving to a multi-node cluster shifts *fed-agent* into another operation node, causing a large increase in overhead. Read and write overhead increase to 3ms and 10ms, respectively. Each node beyond the second adds roughly 0.5–1ms overhead for both reads and writes. The number of messages to reach consensus increases with cluster size, increasing the overhead.

The main component of write overhead is the persistence of transaction logs. A single-node *fed-agent* cluster persists logs to the local disk only and does not need to send them via the network for consensus, leading to a much lower overhead.

The main component of read overhead is transaction conflict detection. Reading transaction status in a single-node cluster is done by directly accessing the local storage. How-



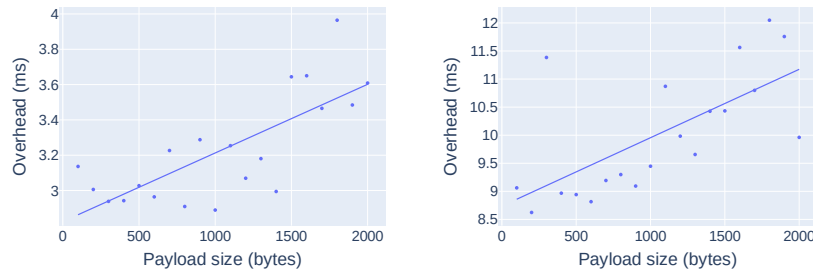
(a) Write overhead, increasing cluster size (b) Read overhead, increasing cluster size

Fig. 11. The effect of cluster size on write and read overhead

ever, reads in multi-cluster deployment need to be *linearizable* [69] to ensure that the previously made changes are applied first. This is necessary to provide high consistency, but it adds an overhead that increases with cluster size. We believe that a weaker consistency level could suffice and would noticeably lower the overhead.

In Fig. 12 we show the effects of payload scaling on write operations. Network communication is needed to reach a consensus on the new transaction state. Transaction status update message contains the original payload, hence the potential impact of payload size on performance. Reads do not update the transaction state, so they are left out of the benchmarks. Payload is scaled to 2kB because most HTTP REST JSON API implementations appear to have payload sizes of 1–2kB, with the median of around 1500B [62].

In both single-node and three-node clusters scenarios, we show the trendline due to the stochastic nature of network latency. For a single-node cluster, write overhead increases by 0.7ms. The increase is attributed to proxying a larger request and the extra time needed to store the payload in the local storage. There is no additional network communication in the single-node cluster, hence a low overhead. The three-node cluster needs to reach a consensus for each write, adding up to 3ms of additional overhead.



(a) Write overhead in a single-node cluster (b) Write overhead in a three-node cluster

Fig. 12. The effect of payload size on write overhead

In conclusion, the overhead of a single-node *fed-agent* cluster for a typical MSA application is expected to be 0.5ms for reads and 2ms for writes. We believe that the overhead can be negligible in an application with usual response times of tens of milliseconds. A multi-node cluster offers higher availability and read parallelism, but increases the overhead to 3ms for reads and 10ms for writes.

5.5. Isolation Level

Fed-agent implements the MVOCC scheme as specified by the SPTM approach. Proper implementation of said scheme should lead to the *SNAPSHOT* isolation. As we are not aware of any tool for formal validation of isolation levels, we implement a test suite attempting to cause isolation anomalies [6]. If anomalies do not occur, we consider the system to be operating at the corresponding isolation level. *Read Skew* and *Write Skew* anomalies are not considered because *fed-agent* supports a limited set of predicate-based operations. *Phantom Read* is the exception and could still occur, as shown in the tests. *Cursor Lost Update* is not considered because *fed-agent* has no notion of cursors. The tests are defined as follows:

- **P0 Dirty Write:** this phenomenon occurs when a transaction overwrites a value written by another active transaction. The test attempts to manifest the anomaly by starting transactions *T1* and *T2* simultaneously, each updating the same user with a different value. *T1* commits before *T2*, which leads to *T2* getting aborted to maintain consistency.
- **P1 Dirty Read:** this phenomenon occurs when a transaction reads a value written by another active transaction. The test attempts to manifest the anomaly by starting transactions *T1* and *T2* simultaneously, each updating the same user with a different value. Both *T1* and *T2* update the user before reading the new value. Both *T1* and *T2* only see the value they modified, meaning that *Dirty Read* did not occur.
- **P2 Non-Repeatable Read:** consider a transaction *T* that reads a value before and after it is modified by another transaction. If *T* reads a different result both times, *Non-Repeatable Read* has occurred. The test attempts to manifest the anomaly by starting transactions *T1* and *T2* simultaneously. *T1* reads the value of a user before and after *T2* updates it. Since both reads yielded the same result, *Non-Repeatable Read* did not occur.
- **P3 Phantom Read:** consider a scenario in which a transaction does a predicate-based read, and another transaction does a write that adds a DO to the result set. If the predicate-based read is repeated and it captures the new value, *Phantom Read* has occurred. The test attempts to manifest the anomaly by starting transactions *T1* and *T2* simultaneously. *T1* reads all skins for a user, which is a predicate-based search. *T2* then updates values for one of the skins and adds a new skin to the set. *T1* then repeats the original read. Since the result stayed the same, *Phantom Read* did not occur.
- **P4 Lost Update:** this phenomenon occurs if active transactions read and modify the same value. The test attempts to manifest the anomaly by starting transactions *T1* and *T2* simultaneously. Both *T1* and *T2* read user credit and increase it by a fixed value. *T1* commits before *T2*, which leads to *T2* getting aborted to maintain consistency.

We ensure that the tests are deterministic by artificially pausing a transaction execution when a state in which an isolation anomaly can happen is reached. Then, another

transaction that attempts to cause a data anomaly executes and the state is checked. We also define tests with invariants that would be violated if data anomalies happened. For example, the total sum of funds across all bank accounts should stay constant before and after the test run, or state of a product must not be a negative number.

5.6. Threats to validity

In this subsection, we briefly discuss the threats to the validity of the results of our research. The biggest threats are those related to the implementation of SPTM, 2PC, and Saga in the Evaluation section. We identify the following:

- 2PC and Saga implementations were specifically built for the purpose of evaluation of *fed-agent*. While it was built with great care, there could be another way to implement 2PC or Saga which would lead to a different evaluation outcome.
- The choice of messaging technology is an important one when implementing Saga. We chose RabbitMQ as it is, in our judgment, a good representative of how a Saga would be implemented in practice due to its popularity. Other message queue technologies, such as ZeroMQ [25], could lead to a lower execution time. However, we believe that the characteristics of Saga, such as a high number of compensations under heavy contention, ultimately have a bigger impact the benchmark results than a choice of technology within the same family.
- The evaluation was made on a specific use case: an e-commerce application. We acknowledge that there are different use cases that have different performance profiles, and thus universal claims cannot be extrapolated from a single comparison.
- The evaluation was not made on a real production system, but on a system built to simulate one. Production systems can be expected to be much more complex, and evaluations within them could lead to different outcomes.
- The evaluation was done on a public cloud provider, namely Amazon Web Services (AWS). This takes away control from us over variables such as network availability and resource contention.

6. Limitations and Future Work

SPTM relies on messages containing enough information to keep track of DO values throughout transaction execution. To our knowledge, there is no rule enforcing the message schema in any API style. What can be defined are the levels of information availability:

1. Level 1 — Messages contain neither DO nor any other identifying information. For example, "count recently added items".
2. Level 2 — Messages do not contain DO but contain identifying information. For example, an increment endpoint for a product stock that returns an empty response.
3. Level 3 — Messages contain partial or convertible DO. For example, an update endpoint that takes or returns partial DO information, or a search endpoint that returns a modified DO.
4. Level 4 — Messages contain the full DO. Single DO format is used for all endpoints handling a type.

SPTM can handle level 1 endpoints if they behave as materialized views. These endpoints can, in some cases, provide enough additional information and auxiliary endpoints for SPTM to estimate the state. Consider a “top 10 selling products” endpoint. A service can provide the count for 10+N products, where N is the buffer in case one of the items of lower placement enters the top 10. Additionally, the service provides a rule to increment the appropriate top 10 record on the “buy” endpoint hit. For more complex rules, snapshotting of endpoints on each DO modification is necessary. This means that in the previous example, the “buy” endpoint would cause SPTM to re-fetch DOs from “top 10 selling products”.

Handling level 2 endpoints is possible if a corresponding read endpoint is provided. In that case, SPTM can track reads on DOs and add them to the version chain. Detected DOs can be used to allow subsequent reads to see a consistent state. If no DO version is known, it is pre-fetched before proceeding with the transaction.

This approach can also be applied to level 3. Alternatively, a conversion algorithm is provided to SPTM, either via a Domain Specific Language (DSL) or a plug-in system. Level 4 endpoints are fully compatible with SPTM and require no modifications.

We expect level 3 and level 4 endpoints to be the most common for OLTP workloads, as it is characterized by short-lived operations on clearly identifiable DOs. Level 1 endpoints serve to connect OLTP results with OLAP data stores, so they are also expected to appear quite commonly. We conjecture that level 2 endpoints could be considered bad practice, as they do not align with any of the major API styles, such as REST. Therefore, we believe that supporting level 1 and level 3 endpoints should be prioritized in the future work before adding support for level 2 endpoints.

In this article, we present an implementation that only considers Option 1 from Section 3.4 for distributed state management. Furthermore, it can only scale reads horizontally, and not writes. In the future, we intend to explore Options 2 and 3 and see how writes can be scaled horizontally within SPTM. We intend to do so by using Conflict-free Replicated Data Types (CRDT) [64] and the MVOCC approach of private transaction workspaces [49]. Transactions are assigned to an SPTM node with a hashing function at the start, which will execute all transaction-related operations in-memory. Messages are sent over the network only once a transaction is finished, minimizing the total number of messages.

Because SPTM operates at the service communication layer, it does not prevent services from accessing uncommitted DO values by directly accessing a data store. We see three potential solutions:

- Service endpoints can be defined as microtransactions: each endpoint can read or write only DOs identifiable from the request or the response. A static analysis tool examining the source code and suggesting changes to fit this criteria can be implemented and provided to developers. However, we recognize that this can be quite restrictive for some systems.
- A service can explicitly ask SPTM what DOs are visible to a transaction. This adds additional overhead and has the potential to violate non-invasiveness property of SPTM.
- Libraries for multiple languages can be developed that inject active values into the persistence layer of a service. This approach should be the least invasive out of the three if properly implemented. The main drawback is that libraries need to be implemented for multiple target languages.

Having a low-to-no impact on existing systems is an important goal of SPTM. In the future, we will be looking into reducing the usage overhead by generating configuration files from existing code or documentation. The first step in this direction would be generating configuration files from language-agnostic documentation specifications, namely Swagger and OpenAPI. Implementing language-specific support for a choice of popular programming languages would come after that.

7. Conclusion

In this article, we introduce the SPTM, a transaction handling approach that supports ACID transactions for OLTP workloads in MSA. A transaction manager implemented in the SPTM approach is easy to use and has a very low impact on code or architecture.

We compare an implementation of SPTM, called *fed-agent*, to 2PC and Saga implementations in a lab-grown e-commerce system. The results show that *fed-agent* outperforms the 2PC implementation by a factor of two for low-medium contention levels. It falls short only in extremely high contention scenarios. As the contention increased in our benchmarks, the 2PC implementation ran into locking bottlenecks that stopped the system from accepting new transactions. *Fed-agent* did not run into this issue and would continue to work beyond the bottleneck point of 2PC. The comparison with Saga shows that *fed-agent* achieves similar performance, while providing high consistency. We also measure the overhead of *fed-agent* and show that it can be negligible for MSA applications with typical response times of tens of milliseconds.

We provide empirical evidence showing that *fed-agent* supports ACID transactions at the *SNAPSHOT ISOLATION* level, which is on par with 2PC using a typical relational database. We believe that having transparent ACID transactions improves the development of MSA in two major ways. First, developers do not need in-depth knowledge of how distributed transactions work to effectively implement them. Second, having ACID transactions for both synchronous and asynchronous message processing helps with data consistency, primarily by reducing the amount of data issues caused by concurrent execution.

References

1. Adya, A., Liskov, B., O’Neil, P.: Generalized isolation level definitions. In: Proceedings of 16th International Conference on Data Engineering (Cat. No.00CB37073). pp. 67–78. IEEE Comput. Soc, San Diego, CA, USA (2000), <http://ieeexplore.ieee.org/document/839388/>
2. Akkoorath, D.D., Tomsic, A.Z., Bravo, M., Li, Z., Crain, T., Bieniusa, A., Pregoica, N., Shapiro, M.: Cure: Strong Semantics Meets High Availability and Low Latency. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS). pp. 405–414. IEEE, Nara, Japan (Jun 2016), <http://ieeexplore.ieee.org/document/7536539/>
3. Arora, V., Nawab, F., Agrawal, D., Abbadi, A.E.: Typhon: Consistency Semantics for Multi-Representation Data Processing. In: 2017 IEEE 10th International Conference on Cloud Computing (CLOUD). pp. 648–655. IEEE, Honolulu, CA, USA (June 2017), <http://ieeexplore.ieee.org/document/8030645/>

4. Bailis, P., Davidson, A., Fekete, A., Ghodsi, A., Hellerstein, J.M., Stoica, I.: Highly available transactions: virtues and limitations. *Proceedings of the VLDB Endowment* 7(3), 181–192 (November 2013), <https://dl.acm.org/doi/10.14778/2732232.2732237>
5. Bailis, P., Ghodsi, A., Hellerstein, J.M., Stoica, I.: Bolt-on causal consistency. In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. pp. 761–772. ACM, New York New York USA (June 2013), <https://dl.acm.org/doi/10.1145/2463676.2465279>
6. Berenson, H., Bernstein, P., Gray, J., Melton, J., O’Neil, E., O’Neil, P.: A critique of ANSI SQL isolation levels. In: *Proceedings of the 1995 ACM SIGMOD international conference on Management of data - SIGMOD ’95*. pp. 1–10. ACM Press, San Jose, California, United States (1995), <http://portal.acm.org/citation.cfm?doid=223784.223785>
7. Bernstein, P.A., Goodman, N.: Multiversion concurrency control—theory and algorithms. *ACM Transactions on Database Systems* 8(4), 465–483 (December 1983), <https://dl.acm.org/doi/10.1145/319996.319998>
8. Bernstein, P.A., Hadzilacos, V., Goodman, N.: *Concurrency control and recovery in database systems*. Addison-Wesley Pub. Co, Reading, Mass (1987)
9. Bogner, J., Fritsch, J., Wagner, S., Zimmermann, A.: *Microservices in Industry: Insights into Technologies, Characteristics, and Software Quality*. In: *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. pp. 187–195. IEEE, Hamburg, Germany (March 2019), <https://ieeexplore.ieee.org/document/8712375/>
10. Braubach, L., Jander, K., Pokahr, A.: A novel distributed registry approach for efficient and resilient service discovery in megascale distributed systems. *Computer Science and Information Systems* 15(3), 751–774 (2018), <http://www.doiserbia.nb.rs/Article.aspx?ID=1820-02141800030B>
11. Conrad, S., Eaglestone, B., Hasselbring, W., Roantree, M., Schöhoff, M., Strässler, M., Vermeer, M., Saltor, F.: Research issues in federated database systems: report of EFDBS ’97 workshop. *ACM SIGMOD Record* 26(4), 54–56 (December 1997), <https://dl.acm.org/doi/10.1145/271074.271089>
12. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with YCSB. In: *Proceedings of the 1st ACM symposium on Cloud computing*. pp. 143–154. ACM, Indianapolis Indiana USA (June 2010), <https://dl.acm.org/doi/10.1145/1807128.1807152>
13. Cowling, J., Liskov, B.: Granola: Low-Overhead Distributed Transaction Coordination. In: *2012 USENIX Annual Technical Conference (USENIX ATC 12)*. pp. 223–235. USENIX Association, Boston, MA (June 2012), <https://www.usenix.org/conference/atc12/technical-sessions/presentation/cowling>
14. Dey, A., Fekete, A., Rohm, U.: Scalable distributed transactions across heterogeneous stores. In: *2015 IEEE 31st International Conference on Data Engineering*. pp. 125–136. IEEE, Seoul, South Korea (April 2015), <http://ieeexplore.ieee.org/document/7113278/>
15. Docs: Amazon ec2 instance types. <https://aws.amazon.com/ec2/instance-types/t3/>, accessed: 2022-05-09
16. Docs: Cassandra documentation. <https://cassandra.apache.org/doc/latest/>, accessed: 2022-05-09
17. Docs: Elasticsearch documentation. <https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html>, accessed: 2022-05-09
18. Docs: The go programming language. <https://go.dev/doc/>, accessed: 2022-05-09
19. Docs: GraphQL documentation. <https://graphql.org/>, accessed: 2022-05-09
20. Docs: grpc documentation. <https://grpc.io/>, accessed: 2022-05-09
21. Docs: Open api specification. <https://spec.openapis.org/oas/latest.html>, accessed: 2022-05-09
22. Docs: PostgreSQL 13. <https://www.postgresql.org/docs/13/index.html>, accessed: 2022-05-09

23. Docs: Protocol buffers specification. <https://developers.google.com/protocol-buffers/docs/overview>, accessed: 2022-05-09
24. Docs: Rabbitmq. <https://www.rabbitmq.com/>, accessed: 2022-04-16
25. Docs: Zeromq. <https://zeromq.org/>, accessed: 2022-04-16
26. Dürr, K., Lichtenthaler, R., Wirtz, G.: An Evaluation of Saga Pattern Implementation Technologies. In: CEUR workshop proceedings. pp. 74–82 (2021), <https://fis.uni-bamberg.de/handle/uniba/49721>, iSSN: 1613-0073 Issue: 2839
27. Fan, G., Chen, L., Yu, H., Qi, W.: Multi-objective optimization of container-based microservice scheduling in edge computing. *Computer Science and Information Systems* 18(1), 23–42 (2021), <http://www.doiserbia.nb.rs/Article.aspx?ID=1820-02142000041F>
28. Fan, P., Liu, J., Yin, W., Wang, H., Chen, X., Sun, H.: 2PC*: a distributed transaction concurrency control protocol of multi-microservice based on cloud computing platform. *Journal of Cloud Computing* 9(1), 40 (December 2020), <https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-020-00183-w>
29. Fielding, R.T.: Architectural Styles and the Design of Network-Based Software Architectures. PhD Thesis, University of California, Irvine (2000), ISBN: 0599871180
30. Fowler, M.: Polyglot persistence. M. Fowler website, [Online]. Available: <https://www.martinfowler.com/bliki/PolyglotPersistence.html> (accessed May 2022)
31. Gadepally, V., Chen, P., Duggan, J., Elmore, A., Haynes, B., Kepner, J., Madden, S., Mattson, T., Stonebraker, M.: The BigDAWG polystore system and architecture. In: 2016 IEEE High Performance Extreme Computing Conference (HPEC). pp. 1–6. IEEE, Waltham, MA, USA (September 2016), <http://ieeexplore.ieee.org/document/7761636/>
32. Github: eshoponcontainers github repository. <https://github.com/dotnet-architecture/eShopOnContainers>, accessed: 2022-05-09
33. Github: Event stream processing microservices github repository. <https://github.com/kbastani/event-stream-processing-microservices>, accessed: 2022-05-09
34. Github: Ftgo github repository. <https://github.com/microservices-patterns/ftgo-application>, accessed: 2022-05-09
35. Github: Lakeside mutual github repository. <https://github.com/Microservice-API-Patterns/LakesideMutual>, accessed: 2022-05-09
36. Github: Sentilo platform github repository. <https://github.com/sentilo/sentilo>, accessed: 2022-05-09
37. Github: Sock shop microservices demo repository. <https://github.com/microservices-demo/microservices-demo>, accessed: 2022-05-09
38. Github: Spring petclinic github repository. <https://github.com/spring-petclinic/spring-petclinic-microservices>, accessed: 2022-05-09
39. Github: Vert.x blueprint project. <https://github.com/sczyh30/vertx-blueprint-microservice>, accessed: 2022-05-09
40. Hasselbring, W., Steinacker, G.: Microservice Architectures for Scalability, Agility and Reliability in E-Commerce. In: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW). pp. 243–246. IEEE, Gothenburg, Sweden (April 2017), <http://ieeexplore.ieee.org/document/7958496/>
41. Helland, P.: Life beyond distributed transactions. *Communications of the ACM* 60(2), 46–54 (January 2017), <https://dl.acm.org/doi/10.1145/3009826>
42. Helland, P.: Data on the outside versus data on the inside. *Communications of the ACM* 63(11), 111–118 (October 2020), <https://dl.acm.org/doi/10.1145/3410623>
43. Junqueira, F., Reed, B., Yabandeh, M.: Lock-free transactional support for large-scale storage systems. In: 2011 IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops (DSN-W). pp. 176–181. IEEE, Hong Kong, China (June 2011), <http://ieeexplore.ieee.org/document/5958809/>

44. Kallman, R., Kimura, H., Natkins, J., Pavlo, A., Rasin, A., Zdonik, S., Jones, E.P., Madden, S., Stonebraker, M., Zhang, Y., others: H-store: a high-performance, distributed main memory transaction processing system. *Proceedings of the VLDB Endowment* 1(2), 1496–1499 (2008), publisher: VLDB Endowment
45. Knoche, H., Hasselbring, W.: Drivers and Barriers for Microservice Adoption – A Survey among Professionals in Germany. *Enterprise Modelling and Information Systems Architectures (EMISAJ)* pp. 1:1–35 Pages (January 2019), <https://emisa-journal.org/emisa/article/view/164>
46. Krylovskiy, A., Jahn, M., Patti, E.: Designing a Smart City Internet of Things Platform with Microservice Architecture. In: *2015 3rd International Conference on Future Internet of Things and Cloud*. pp. 25–30. IEEE, Rome, Italy (August 2015), <https://ieeexplore.ieee.org/document/7300793/>
47. Kulkarni, S.S., Demirbas, M., Madappa, D., Avva, B., Leone, M.: Logical Physical Clocks. In: Aguilera, M.K., Querzoni, L., Shapiro, M. (eds.) *Principles of Distributed Systems*, vol. 8878, pp. 17–32. Springer International Publishing, Cham (2014), series Title: *Lecture Notes in Computer Science*
48. Laigner, R., Zhou, Y., Salles, M.A.V., Liu, Y., Kalinowski, M.: Data management in microservices: state of the practice, challenges, and research directions. *Proceedings of the VLDB Endowment* 14(13), 3348–3361 (September 2021), <https://dl.acm.org/doi/10.14778/3484224.3484232>
49. Larson, P.A., Blanas, S., Diaconu, C., Freedman, C., Patel, J.M., Zwilling, M.: High-performance concurrency control mechanisms for main-memory databases. *Proceedings of the VLDB Endowment* 5(4), 298–309 (December 2011), <https://dl.acm.org/doi/10.14778/2095686.2095689>
50. Levandoski, J.J., Lomet, D.B., Mokbel, M.F., Zhao, K.: Deuteronomy: Transaction Support for Cloud Data. In: *Fifth Biennial Conference on Innovative Data Systems Research, CIDR 2011, Asilomar, CA, USA, January 9-12, 2011, Online Proceedings*. pp. 123–133. [www.cidrdb.org \(2011\), http://cidrdb.org/cidr2011/Papers/CIDR11_Paper14.pdf](http://cidrdb.org/cidr2011/Papers/CIDR11_Paper14.pdf)
51. Li, J., Lu, Y., Zhang, Y., Wang, Q., Cheng, Z., Huang, K., Shu, J.: SwitchTx: scalable in-network coordination for distributed transaction processing. *Proceedings of the VLDB Endowment* 15(11), 2881–2894 (July 2022), <https://dl.acm.org/doi/10.14778/3551793.3551838>
52. Limon, X., Guerra-Hernandez, A., Sanchez-Garcia, A.J., Perez Arriaga, J.C.: SagaMAS: A Software Framework for Distributed Transactions in the Microservice Architecture. In: *2018 6th International Conference in Software Engineering Research and Innovation (CONISOFT)*. pp. 50–58. IEEE, San Luis Potosí, Mexico (October 2018), <https://ieeexplore.ieee.org/document/8645853/>
53. Lotz, J., Vogelsang, A., Benderius, O., Berger, C.: Microservice Architectures for Advanced Driver Assistance Systems: A Case-Study. In: *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. pp. 45–52. IEEE, Hamburg, Germany (March 2019), <https://ieeexplore.ieee.org/document/8712376/>
54. Lykhenko, T., Soares, R., Rodrigues, L.: FaaSTCC: efficient transactional causal consistency for serverless computing. In: *Proceedings of the 22nd International Middleware Conference*. pp. 159–171. ACM, Québec city Canada (December 2021), <https://dl.acm.org/doi/10.1145/3464298.3493392>
55. M. Del Esposte, A., Kon, F., M. Costa, F., Lago, N.: InterSCity: A Scalable Microservice-based Open Source Platform for Smart Cities. In: *Proceedings of the 6th International Conference on Smart Cities and Green ICT Systems*. pp. 35–46. SCITEPRESS - Science and Technology Publications, Porto, Portugal (2017), <http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006306200350046>
56. Mattern, F.: *Virtual time and global states of distributed systems*. Univ., Department of Computer Science (1988)

57. Mazzara, M., Dragoni, N., Bucchiarone, A., Giaretta, A., Larsen, S.T., Dustdar, S.: Microservices: Migration of a Mission Critical System. *IEEE Transactions on Services Computing* 14(5), 1464–1477 (September 2021), <https://ieeexplore.ieee.org/document/8585089/>
58. Ongaro, D., Ousterhout, J.: In Search of an Understandable Consensus Algorithm. In: 2014 USENIX Annual Technical Conference (USENIX ATC 14). pp. 305–319. USENIX Association, Philadelphia, PA (June 2014), <https://www.usenix.org/conference/atc14/technical-sessions/presentation/ongaro>
59. Patiño-Martinez, M., Jiménez-Peris, R., Kemme, B., Alonso, G.: MIDDLE-R: Consistent database replication at the middleware level. *ACM Transactions on Computer Systems* 23(4), 375–423 (November 2005), <https://dl.acm.org/doi/10.1145/1113574.1113576>
60. Ports, D.R.K., Grittner, K.: Serializable snapshot isolation in PostgreSQL. *Proceedings of the VLDB Endowment* 5(12), 1850–1861 (August 2012), <https://dl.acm.org/doi/10.14778/2367502.2367523>
61. Pritchett, D.: BASE: An Acid Alternative: In partitioned databases, trading some consistency for availability can lead to dramatic improvements in scalability. *Queue* 6(3), 48–55 (May 2008), <https://dl.acm.org/doi/10.1145/1394127.1394128>
62. Rodríguez, C., Baez, M., Daniel, F., Casati, F., Trabucco, J.C., Canali, L., Percannella, G.: REST APIs: A Large-Scale Analysis of Compliance with Principles and Best Practices. In: Bozzon, A., Cudre-Maroux, P., Pautasso, C. (eds.) *Web Engineering*, vol. 9671, pp. 21–39. Springer International Publishing, Cham (2016), https://link.springer.com/10.1007/978-3-319-38791-8_2, series Title: Lecture Notes in Computer Science
63. Rudrabhatla, C.K.: Comparison of Event Choreography and Orchestration Techniques in Microservice Architecture. *International Journal of Advanced Computer Science and Applications* 9(8) (2018), <http://thesai.org/Publications/ViewPaper?Volume=9&Issue=8&Code=ijacsa&SerialNo=4>
64. Shapiro, M., Preguiça, N., Baquero, C., Zawirski, M.: Conflict-Free Replicated Data Types. In: Défago, X., Petit, F., Villain, V. (eds.) *Stabilization, Safety, and Security of Distributed Systems*, vol. 6976, pp. 386–400. Springer Berlin Heidelberg, Berlin, Heidelberg (2011), http://link.springer.com/10.1007/978-3-642-24550-3_29, series Title: Lecture Notes in Computer Science
65. Sheth, A.P., Larson, J.A.: Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys* 22(3), 183–236 (September 1990), <https://dl.acm.org/doi/10.1145/96602.96604>
66. Soldani, J., Tamburri, D.A., Van Den Heuvel, W.J.: The pains and gains of microservices: A Systematic grey literature review. *Journal of Systems and Software* 146, 215–232 (December 2018), <https://linkinghub.elsevier.com/retrieve/pii/S0164121218302139>
67. Thomson, A., Diamond, T., Weng, S.C., Ren, K., Shao, P., Abadi, D.J.: Calvin: fast distributed transactions for partitioned database systems. In: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. pp. 1–12. ACM, Scottsdale Arizona USA (May 2012), <https://dl.acm.org/doi/10.1145/2213836.2213838>
68. Viennot, N., Lécuyer, M., Bell, J., Geambasu, R., Nieh, J.: Synapse: a microservices architecture for heterogeneous-database web applications. In: *Proceedings of the Tenth European Conference on Computer Systems*. pp. 1–16. ACM, Bordeaux France (April 2015), <https://dl.acm.org/doi/10.1145/2741948.2741975>
69. Viotti, P., Vukolić, M.: Consistency in Non-Transactional Distributed Storage Systems. *ACM Computing Surveys* 49(1), 1–34 (March 2017), <https://dl.acm.org/doi/10.1145/2926965>

70. Wei, Z., Pierre, G., Chi, C.H.: CloudTPS: Scalable Transactions for Web Applications in the Cloud. *IEEE Transactions on Services Computing* 5(4), 525–539 (2012), <http://ieeexplore.ieee.org/document/5740834/>
71. Wu, C., Sreekanti, V., Hellerstein, J.M.: Transactional Causal Consistency for Serverless Computing. In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. pp. 83–97. ACM, Portland OR USA (June 2020), <https://dl.acm.org/doi/10.1145/3318464.3389710>
72. Wu, Y., Arulraj, J., Lin, J., Xian, R., Pavlo, A.: An empirical evaluation of in-memory multi-version concurrency control. *Proceedings of the VLDB Endowment* 10(7), 781–792 (March 2017), <https://dl.acm.org/doi/10.14778/3067421.3067427>
73. Zhang, G., Ren, K., Ahn, J.S., Ben-Romdhane, S.: GRIT: Consistent Distributed Transactions Across Polyglot Microservices with Multiple Databases. In: *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. pp. 2024–2027. IEEE, Macao, Macao (April 2019), <https://ieeexplore.ieee.org/document/8731442/>

Lazar Nikolić received his M.Sc degree from the Faculty of Technical Sciences, at the University of Novi Sad in 2016. In the same year, he enrolled in the Ph.D. program at the Faculty of Technical Sciences, at the University of Novi Sad. He was a teaching assistant until early 2023 at the Faculty of Technical Sciences, at the University of Novi Sad, where he participated in several courses in the area of Web Programming and Internet Networks. He is currently working on his Ph.D. thesis in the area of Distributed Systems, Microservice Architectures, and Transaction Management Systems.

Vladimir Dimitrieski works as an associate professor at the University of Novi Sad, Faculty of Technical Sciences, Serbia. There, he went through all the academic education levels, receiving a Ph.D. in computer science in 2018. Currently, he is a lecturer in several courses at the Faculty of Technical Sciences that cover domains of (meta-)modeling, domain-specific languages, and data engineering. With a strong background in the domain of Industry 4.0, (meta-)modeling and data engineering, he has been part of multiple national, international, and industrial projects in these domains.

Milan Čeliković received his M.Sc. degree from the Faculty of Technical Sciences, at the University of Novi Sad in 2009. He received his Ph.D. degree in 2018, at the University of Novi Sad, Faculty of Technical Sciences. Currently, he works as an assistant professor at the Faculty of Technical Sciences at the University of Novi Sad, where he lectures several Computer Science and Informatics courses. His main research interests are focused on: Databases, Database management systems, Information Systems, and Software Engineering.

Received: December 10, 2022; Accepted: November 22, 2023.

SPC5: an efficient SpMV framework vectorized using ARM SVE and x86 AVX-512

Evann Regnault¹ and Bérenger Bramas^{2,3}

¹ Strasbourg University, UFR de Mathématique et d'Informatique
7, rue René Descartes, 67084 Strasbourg, France
evann.regnault@etu.unistra.fr

² Inria Nancy, CAMUS Team, 615 Rue du Jardin-Botanique
54600 Villers-lès-Nancy, France

³ ICube laboratory, ICPS Team, 300 bd Sébastien Brant
67412 Illkirch Cedex, France
berenger.bramas@inria.fr

Abstract. The sparse matrix/vector product (SpMV) is a fundamental operation in scientific computing. Having access to an efficient SpMV implementation is therefore critical, if not mandatory, to solve challenging numerical problems. The ARM-based AFX64 CPU is a modern hardware component that equips one of the fastest supercomputers in the world. This CPU supports the Scalable Vector Extension (SVE) vectorization technology, which has been less investigated than the classic x86 instruction set architectures. In this paper, we describe how we ported the SPC5 SpMV framework on AFX64 by converting AVX512 kernels to SVE. In addition, we present performance results by comparing our kernels against a standard CSR kernel for both Intel-AVX512 and Fujitsu-ARM-SVE architectures.

Keywords: SpMV, vectorization, AVX-512, SVE.

1. Introduction

The sparse matrix/vector product (SpMV) is a fundamental operation in scientific computing. It is the most important component of iterative linear solvers, which are widely used in finite element solvers. This is why SpMV has been and remains studied and improved.

Most of the studies work on the storage of sparse matrices, the implementation of SpMV kernels for novel hardware, or the combination of both.

In a previous work [7], we proposed a new sparse matrix storage format and its corresponding SpMV kernel in a framework called SPC5. The implementation was for x86 CPUs using the AVX512 instruction set architectures, and it was efficient for various types of data distribution.

In the current work, we are interested in porting this implementation on ARM SVE [21,4,3] architecture. In other words, we aim at keeping the SPC5 storage format but create computational kernels that are efficient on ARM CPUs with SVE.

AVX512 and SVE instruction set architectures are different in their philosophies and features. Consequently, as it is usually the case with vectorization, providing a new computational kernel is like solving a puzzle: we have the operation we want to perform on one side and the existing hardware instructions on the other side.

The contribution of the paper is to depict a new SpMV kernel for ARM SVE, and to demonstrate its performance on several sparse matrices of different shapes. A secondary contribution is the description of our new AVX512 implementation, which is much simpler than the previous assembly implementation, while still delivering the same performance.

This paper is organized as follows. In Section 2, we start by describing the vectorization principle, then the SpMV operation and the challenges of its efficient implementation, and finally provide the specificities of SPC5. Then, in Section 2.4, we present our SPC5 implementation with SVE. Finally, we study the performance of our implementation in Section 4.

2. Background

2.1. Vectorization

Vectorization, also named SIMD for single instruction multiple data [11], is a key mechanism of modern processing units to increase the performance despite the clock frequency stagnation. As its name suggests, the idea consists in working on several elements stored in vectors instead of scalar distinct elements. As such, instead of performing operations on one element at a time, we perform the operations on vectors of elements using a vector instruction set architecture (ISA) that supports vector instructions. We provide a schematic view of the concept in Figure 1.

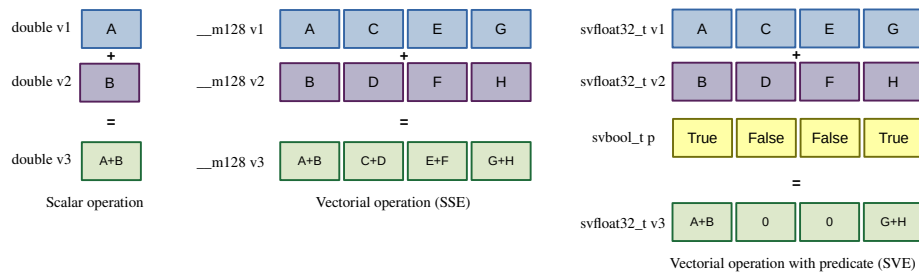


Fig. 1. Illustration of a scalar operation, a vectorial operation and a vectorial operation with predicate. Each of the three instructions is performed with a single instruction

Vectorization is straightforward when we aim to apply the same operation on all the elements of a vector. However, the principle is challenging when we have divergence, i.e., we do not apply the exact same operations on all the elements, or when we need to perform data layout transformations, i.e., the input/output data blocks from the main memory that are loaded from (stored to) vectors are not contiguous, or we need to shuffle the data inside the vectors.

Moreover, not all instruction sets support the same operations, making each implementation specific to a given hardware. Consequently, what could be done with a single

instruction in a given instruction set, might need several instructions in another. For example, non-contiguous stores (scatter), non-contiguous loads (gathers), or internal permutation/merging of vectors are not available in all existing instruction sets and not necessarily similar when they are supported.

Many computational algorithms use conditional statements, therefore several solutions have been proposed to manage vector divergences. The first one is the single instruction multiple thread (SIMT) programming model, as used in CUDA and OpenCL. While the programmer expresses its parallel algorithm as if independent execution threads would be used, it is actually large vector units that will perform the execution, where each thread will be an element of the vector. The hardware takes care of the coherency during the execution.

The second mechanism is the use of a vector of predicates, where each predicate tells if an operation should be applied on an element of the vector. When the elements of a vector should follow different execution paths (branches), all paths will be executed but predicate vectors will ensure to apply the correct operations. The ARM SVE technology uses this mechanism, and most instructions can be used with a predicate vector. Similar behavior can be obtained with classic x86 instruction sets using, for example, binary operations to merge several vectors obtained through different execution branches.

2.2. Related Work on Vectorized with SVE

Developing optimized kernels with SVE is a recent research topic [18,14,2,25,10]. A previous study [1] has focused on the modelling and tuning of the A64FX CPU. The authors implemented the SELL-C- σ SpMV kernels and tuned it for this hardware. This kernel was originally made for GPUs but works well on CPUs too. However, the format is very different from the CSR format and requires a costly conversion step, which we aim to avoid. Additionally, the authors have performed important tuning for each matrix, by permuting the matrix or performing costly parameter optimization, where we want to provide a unique solution.

2.3. SpMV

The SpMV operation has been widely studied. This operation is memory bound in most cases with a low arithmetic intensity. Consequently, a naive vectorization usually does not provide significant benefits if the arithmetic intensity remains unchanged. This is why the storage of the sparse matrix is usually the central point of improvement.

Each new ISA can potentially help to create new storage formats that take less memory and/or that can be vectorized more efficiently.

For example, consider the more simple storage format called *coordinates* (COO) or IJV, where each non-zero value (NNZ) is stored with a triple row index, column index and floating point value. In this case, for each NNZ we need two integers and one floating point value. Not only this format is heavy but it is difficult to vectorized its corresponding SpMV kernel.

Another well-known storage format is the *compressed sparse row* (CSR), where the values of the same row are stored contiguously such that there is no need to store an individual row index per value. With the CSR, each NNZ needs a single integer, which is the column index, decreasing the memory footprint up to 33% compared to COO/IJV.

Following this idea, plenty of storage formats have been proposed. Many of them also tried to obtain a format that can be computed efficiently for a given architecture.

Some of the first block-based formats are the block compressed sparse row storage (BCSR) [20] and its extensions to larger blocks of variable dimension [24,13] or to unaligned block compressed sparse row (UBCSR) [23]. However, in these formats, the blocks have to be filled with zeros to be full. For these formats, the blocks were aligned (the upper-left corner of the blocks start at a position multiple of the block size). While the blocks are well suited for vectorization, the extra zeros can dramatically decrease the performance.

More recent work has focused on GPUs and manycore architectures. Among them, the references are the ELLPACK format [17], SELL-C- σ [15] defined as a variant of Sliced ELLPACK, and the CSR5 [16] format that we used as reference in our previous study.

The Cuthill-McKee method from [8] is a well-known technique for improving the bandwidth of a matrix to have good properties for LU decomposition. It does so by applying a breadth-first algorithm on a graph which represents the matrix structure. While the aim of this algorithm is not to improve the SpMV performance, the generated matrices may have better data locality.

Another method [20] has been specifically designed to increase the number of contiguous values in rows and/or columns. This method works by creating a graph from a matrix, where each column (or row) is a vertex and all the vertices are connected with weighted edges. The weights represent the interest of putting two columns (or rows) contiguously. By solving the traveling salesman problem (TSP) to obtain a path that goes through all the nodes but only once and that minimizes the total weight of the path, we can find a permutation of the sparse matrix that should be better divided into blocks. This means that we should have fewer blocks and the blocks should contain more NNZ elements.

Several updates to the method have been presented in [23,19,5] using different formulas. While the current study does not focus on the permutation of matrices, it is worth noting that enhancing the matrix's shape, as in other approaches, would likely lead to improved kernel efficiency by reducing the number of blocks.

2.4. SPC5

The SPC5 format consists in using a block scheme without adding additional zeros. SPC5 can be seen as an extension of the CSR format, but where the values of each row are split into blocks. Each block starts with a NNZ at column c and includes the next NNZ values until column $c+VEC_SIZE-1$ if they exist. Consequently, in the worst case a block contains a single value, and in the best case VEC_SIZE values. Then, for each block, we use a mask of bits to indicate which of the NNZ values in the block exist. As a result, in a poor configuration, SPC5 will have the same memory footprint as the CSR plus one bit mask per NNZ. On the other hand, in the other extreme scenario, SPC5 will save one integer for each value added to a block since we can retrieve the corresponding column index from c and the position of the NNZ in the block.

The SPC5 format has been extended so that a block is mapped to several rows. This is helpful if there are NNZ values closed (NNZ of consecutive rows that have closed column index) such that the values loaded from the vector x can be used more than once and that the column index of the block is reused for more NNZ.

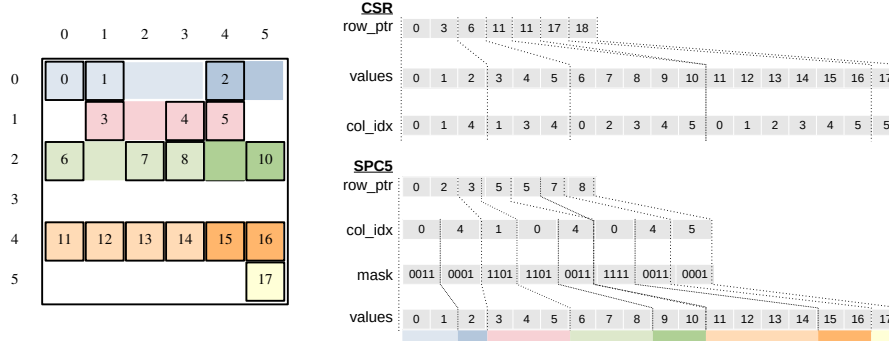


Fig. 2. Illustration of the CSR and SPC5 formats. In this figure, we use the $\beta(1,4)$ format, which means that each block is on a single row and of length 4. In the CSR format, the original *row_idx* array is compacted to have a single index per block instead of an index per NNZ. The mask array indicates the positions of the next NNZ in the block, and the corresponding column index can be obtained by summing the block's column index with the corresponding bit position in the mask

In the rest of the document, we refer to $\beta(r, \text{VEC_SIZE})$ when the blocks are over r rows and of length VEC_SIZE . In the original study, we were also using blocks of $\text{VEC_SIZE}/2$ but not in the current study. We give an example of CSR and SPC5 in Figure 2.

3. SPC5 Implementation

We provide the SPC5 SpMV pseudocode in Algorithm 1.

First, we initialize an index to progress in the array of NNZ values, at line 3. Since we have no way to know where the values of a given block are located in the array, we have to increase the index with the number of values of each block that has been computed. This is visible at line 16 for the scalar version, line 21 for AVX512, and line 28 for SVE.

At line 5, we iterate over the rows with a step r . For each row segment, we iterate over the blocks at line 8. For each block, we load its column index at line 9.

Then, we process each row of the block. We start by getting the mask, at line 11, that tells us what are the NNZ that exist in the row of the block. A naive implementation consists in testing the existence of each possible value, at line 14, and performing the computation if needed, at line 15. However, this loop over the NNZ can be done with a few instructions in AVX512, at line 20. In this case, we load a vector from x that matches the column index, and expand the NNZ from the value array into a vector.

The expand operation does not exist in SVE. Consequently, the same behavior is obtained using different instructions. First, we need a filter vector that contains 2^i at position i , see line 4. Then, we compute a binary *and* operation between the filter vector and the mask, such that only the position for which a NNZ exist will not be zero. We do this operation at line 23 and get the active elements at line 24. Second, instead of expanding

the NNZ values, as with AVX512, we compact the values from x , at line 26. Doing so, we can simply load the right number of NNZ and leave them contiguous in the resulting vector, before performing the computation. A schematic view of the two approaches is provided in Figure 3.

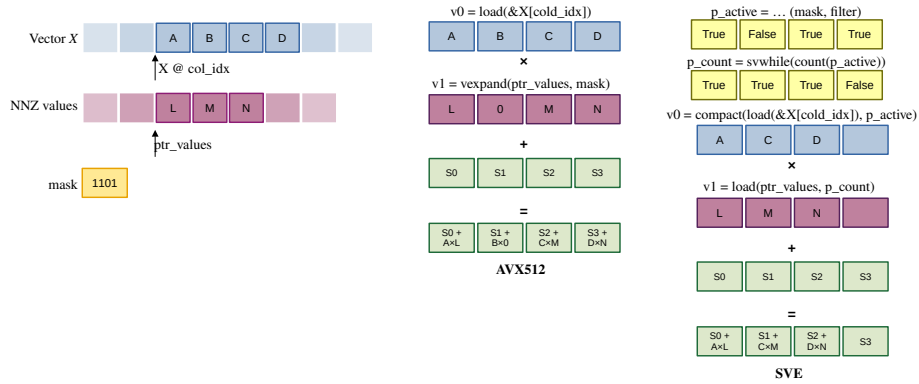


Fig. 3. Illustration of the loading and computation of one row of a block. The mask is represented from the most significant bit (MSB) to the least significant bit (LSB), whereas the vector elements are represented from the first element to the last element. Hence, the 1s in the mask 1101 correspond to the elements N, M, and L (in this order)

Finally, we update y at the end of the algorithm (at line 32) for each of the rows that have been proceeded.

3.1. Optimizing the Loading of x

In AVX512, the values from x are loaded into a SIMD vector without pruning. This means that no matter how many NNZ are in the block or how many values we need from x , VEC.SIZE values will be loaded from memory. It is possible to prune/filter the values, but this will imply an extra cost (i.e., using a more expensive instruction like gather) and would certainly have no benefit as the AVX512 SIMD load instruction is translated into an efficient memory transaction. Consequently, in AVX512, the main optimization consists in loading the values from x once for all the rows of a block, which allows accessing the memory once and using the resulting vector r times.

ALGORITHM 1: SpMV for a matrix mat in format $\beta(r, c)$. The lines in blue • are to compute in scalar and have to be replaced by the line in green • to have the vectorized equivalent in SVE or in red • with AVX

Input: x : vector to multiply with the matrix. mat : a matrix in the block format $\beta(r, c)$. r, c : the size of the blocks.

Output: y : the result of the product.

```

1 function spmv( $x, mat, r, c, y$ )
2   // Index to access the array's values
3    $idxVal \leftarrow 0$ 
4    $filter \leftarrow [1 \lll 0, \dots, 1 \lll VS-1]$ 
5   for  $idxRow \leftarrow 0$  to  $mat.numberOfRows-1$  inc by  $r$  do
6      $sum[r] \leftarrow init\_scalar\_array(r, 0)$ 
7      $sum[r] \leftarrow init\_simd\_array(r, 0)$ 
8      $for$   $idxBlock \leftarrow mat.block\_rowptr[idxRow/r]$  to  $mat.block\_rowptr[idxRow/r+1]-1$ 
9       do
10         $idxCol \leftarrow mat.block\_colidx[idxBlock]$ 
11        for  $idxRowBlock \leftarrow 0$  to  $r$  do
12           $valMask \leftarrow mat.block\_masks[idxBlock \times r + idxRowBlock]$ 
13          // The next loop can be vectorized with vexpand
14          for  $k \leftarrow 0$  to  $c$  do
15            if  $bit\_shift(1, k) BIT\_AND$   $valMask$  then
16               $sum[idxRowBlock] += x[idxCol+k] * mat.values[idxVal]$ 
17               $idxVal += 1$ 
18            end
19          end
20          // To replace the k-loop AVX512
21           $sum[idxRowBlock] += simd\_load(x[idxCol]) *$ 
22           $simd\_vexpand(mat.values[idxVal], valMask)$ 
23           $idxVal += popcount(valMask)$ 
24          // To replace the k-loop SVE
25           $mask\_vec = svand(svdup(valMask), filter)$ 
26           $active\_elts = svcmpne(mask\_vec, 0)$ 
27           $increment = count(active\_elts)$ 
28           $xvals = svcompact(active\_elts, simd\_load(active\_elts, x[idxCol]))$ 
29           $block = simd\_load(svwhile(0, increment), mat.values[idxVal])$ 
30           $idxVal += increment$ 
31           $sum[idxRowBlock] += block * xvals$ 
32        end
33      end
34       $y[idxRowBlock] += sum[r]$ 
35       $y[idxRowBlock] += simd\_hsum(sum[r])$ 
36    end

```

With SVE, it is different and we have mainly three alternatives:

- Loading the values from x without pruning, as with AVX512, and then compacting the obtained vector for each row of the block. We refer to this strategy as *single x load*.
- Loading a different vector for each row of the block, as shown in Algorithm 1. We refer to this strategy as *partial x load*.
- Combining the predicates of several rows of the block by merging their predicates/masks to load all the values that are needed by the block, but not more. In our study, we left this approach aside, as different tests we have conducted have shown poor performance.

The performance gains we can expect from the different strategies depend on the way the load is actually performed by the hardware. In fact, the main point is to know if the hardware can make faster memory transactions when some elements of the predicate vector used in the load are false. If the hardware actually loads `VEC_SIZE` values from the memory but then only copies the ones that have their corresponding predicate value true to the SIMD vector, we should not expect any benefit. Moreover, ARM SVE can be seen as an interface, hence it can be implemented by the hardware differently such that the behavior can also change from one vendor to another.

3.2. Optimizing the Writing of the Result in y

In the SIMD implementation, we have to perform the reduction (i.e., the horizontal sum) of r vectors to add them to y and store the result.

A straightforward approach is to call a single reduction instruction per vector, as both AVX512 and SVE support such an operation. However, this means that we will perform r individual summations between the reduced values and the values from y , and that we will also access the memory r times (actually $2 \times r$, since we load values from y , perform the summation, and write back to y).

To avoid this, we propose a possible optimization that consists in performing the reduction of all the vectors manually to obtain a single vector as output, and then performing a vectorial summation with y .

With AVX512, this manual multi-reduction can be implemented by playing with AVX and SSE registers and using the horizontally add adjacent pairs instruction (*hadd*). The operation is done without any loop.

With SVE, we do this using odd/even interleave instructions (*svuzp1* and *svuzp2*). In this case, we need a loop because the length of the vectors is unknown at compile time.

4. Performance Study

4.1. Configuration

We assess our method on two configurations:

- Fujitsu-SVE: it is an ARMv8.2 A64FX - Fujitsu with 48 cores at 1.8 GHz and 512-bit SVE [12], i.e. a vector can contain 16 single precision floating-point values or 8

double precision floating-point values. The node has 32 GB HBM2 memory arranged in four core memory groups (CMGs) with 12 cores and 8GB each, 64KB private L1 cache, and 8MB shared L2 cache per CMG. We use the GNU compiler 10.3.0.

- Intel-AVX512: it is a 2×18 -core Cascade Lake Intel Xeon Gold 6240 at 2.6 GHz with AVX-512 (Advanced Vector 512-bit, Foundation, Conflict Detection, Byte and Word, Doubleword and Quadword Instructions, and Vector Length). The main memory consists of 190 GB DRAM memory arranged in two NUMA nodes. Each CPU has 18 cores with 32KB private L1 cache, 1024KB private L2 cache, and 25MB shared L3 cache. We use the GNU compiler 11.2.0 and the MKL 2022.0.2.

4.2. Test Cases

We evaluated the performance of our proposed SPC5 SpMV kernels ⁴ on a set of sparse matrices taken from the University of Florida sparse matrix collection (UF Collection) [9]. It includes a dense matrix of dimension 2048. The results of the dense matrix are expected to provide an upper bound on the performance of the kernels, as all blocks will be full. Of course, our kernels are not well-designed or optimized for a dense matrix-vector product. The properties of the matrices are given in Table 1.

We evaluated the performance of four kernels: $\beta(1, VS)$, $\beta(2, VS)$, $\beta(4, VS)$, and $\beta(8, VS)$, where VS is the vector size. We also provide the filling of the blocks when we format the matrices to the corresponding block sizes. The filling can be up to 80% for *nd6k* but as low as 1% for *wikipedia-20060925*. (It is obviously 100% for the dense matrix.)

We performed the computation in single precision (*floatf32*) and double precision (*doubleff64*).

The original AVX implementation was written in assembly language, while our current implementation is written in C++ with intrinsic functions. Consequently, the AVX and SVE kernels have very similar structures.

4.3. Results

The results are organized as follows. In the first part, we evaluate the difference of using the manual multi-reduction (described in Section 3.2) vs. the native SIMD horizontal summation in Figures 2 (a) and 2 (b), for Fujitsu-SVE and Intel-AVX respectively. For Fujitsu-SVE, we also evaluate the use of full vector load of x (described in Section 3.2). Then, we provide the detailed results for all the matrices and the selected configuration in Figures 5 and 7. Finally, we provide an overview of the parallel performance when the computation is naively divided among the threads in Figure 8.

Comparisons of the Different Optimizations. We provide the performance results of the different implementations in Table 2.

In Table 2 (a), we evaluate the use of manual multi-reduction and the single load of the x vector for the Fujitsu-SVE architecture. There is no difference in most cases, and it is always meaningless for $\beta(2, VS)$. The $\beta(4, VS)$ and $\beta(8, VS)$ kernels react differently with the optimizations and improve slightly. The small change in using the multi-reduction comes from the small difference in latencies between the two approaches. The *reduce*

⁴ The code is available at <https://gitlab.inria.fr/bramas/spc5>

Table 1. Matrix set for computation and performance analysis. We provide the percentage of filling of the blocks for double (left) and single (right) precision

Name	Dim	NNZ	$\frac{NNZ}{N_{rows}}$	$\beta(1, VS)$		$\beta(2, VS)$		$\beta(4, VS)$		$\beta(8, VS)$	
bundle	513351	20208051	39.365	72%	55%	70%	54%	64%	50%	51%	46%
CO	221119	7666057	34.6694	18%	9%	18%	9%	17%	9%	16%	8%
crankseg	63838	14148858	221.637	66%	49%	59%	44%	49%	37%	38%	29%
dense	2048	4194304	2048	100%	100%	100%	100%	100%	100%	100%	100%
dielFilterV2real	1157456	48538952	41.9359	31%	20%	22%	14%	15%	10%	11%	7%
Emilia	923136	41005206	44.4195	50%	31%	43%	28%	34%	24%	24%	18%
FullChip	2987012	26621990	8.91258	24%	13%	17%	10%	13%	7%	8%	5%
Hook	1498023	60917445	40.6652	51%	34%	43%	29%	33%	23%	24%	17%
in-2004	1382908	16917053	12.233	48%	31%	38%	25%	30%	19%	21%	14%
ldoor	952203	46522475	48.8577	87%	55%	79%	51%	67%	44%	51%	34%
mixtank	29957	1995041	66.5968	31%	20%	24%	16%	17%	11%	12%	8%
nd6k	18000	6897316	383.184	80%	71%	76%	68%	71%	64%	64%	58%
ns3Da	20414	1679599	82.2768	14%	7%	8%	4%	4%	2%	2%	1%
pdb1HYS	36417	4344765	119.306	77%	65%	72%	60%	63%	54%	54%	46%
pwtk	217918	11634424	53.389	74%	56%	74%	55%	73%	54%	65%	53%
RM07R	381689	37464962	98.1557	61%	41%	51%	34%	40%	28%	31%	25%
Serena	1391349	64531701	46.3807	51%	34%	43%	29%	33%	23%	24%	17%
Si41Ge41H72	185639	15011265	80.8627	32%	18%	31%	17%	28%	15%	22%	13%
Si87H76	240369	10661631	44.3553	21%	11%	21%	11%	20%	10%	17%	9%
spal	10203	46168124	4524.96	74%	69%	45%	37%	25%	23%	13%	12%
torso1	116158	8516500	73.3182	81%	63%	80%	62%	77%	59%	58%	55%
TSOPF	38120	16171169	424.217	94%	88%	93%	87%	92%	85%	89%	82%
wikipedia-20060925	2983494	37269096	12.4918	13%	6%	6%	3%	3%	1%	1%	0%

instruction (*addv*) has a latency of 12 cycles [12]. Our multi-reduction has a latency of around 96 cycles for two vectors (considering the following latencies *uzp1* 6, *uzp2* 6, *whilelt* 4 and full *vadd* 22), and it is almost the same cost for 4 or 8 vectors. Disabling the *x* load optimization almost always degrades performance for the $\beta(4, VS)$ kernel but seems to improve the performance for the $\beta(8, VS)$. This is surprising, as we would expect that the larger the blocks would be, the more benefit we would have to load the vector from *x* completely. From our understanding, the cost of a load depends on the location of the data it requests but not on the fact that the data it requests could be located in different cache lines. This explains why the optimization has a limited impact, as a partial load will move the data to the cache and speedup the next partial loads. Since the $\beta(4, VS)$ is faster than $\beta(8, VS)$, we consider the best configuration to be with both optimizations turned on.

In Table 2 (b), we evaluate the use of manual multi-reduction on Intel-AVX512 architecture. The performance increases slightly with the use of manual multi-reduction in some cases. For instance, the best performance on average is obtained with $\beta(4, VS)$ and for this configuration, using the manual multi-reduction has no impact (double) or increases the speedup by 0.1 (float). The explanation is as follows: the *reduce* intrinsic function (*_mm512_reduce_add_ps*) is not actually a real hardware instruction, but a call to a function provided by the compiler [22]. Its implementation⁵ is very similar to our manual multi-reduction, with the main difference being that we try to factorize the in-

⁵ <https://github.com/gcc-mirror/gcc/blob/9d7e19255c06e05ad791e9bf5aefc4783a12c4f9/gcc/config/i386/avx512fintrin.h#L15928>

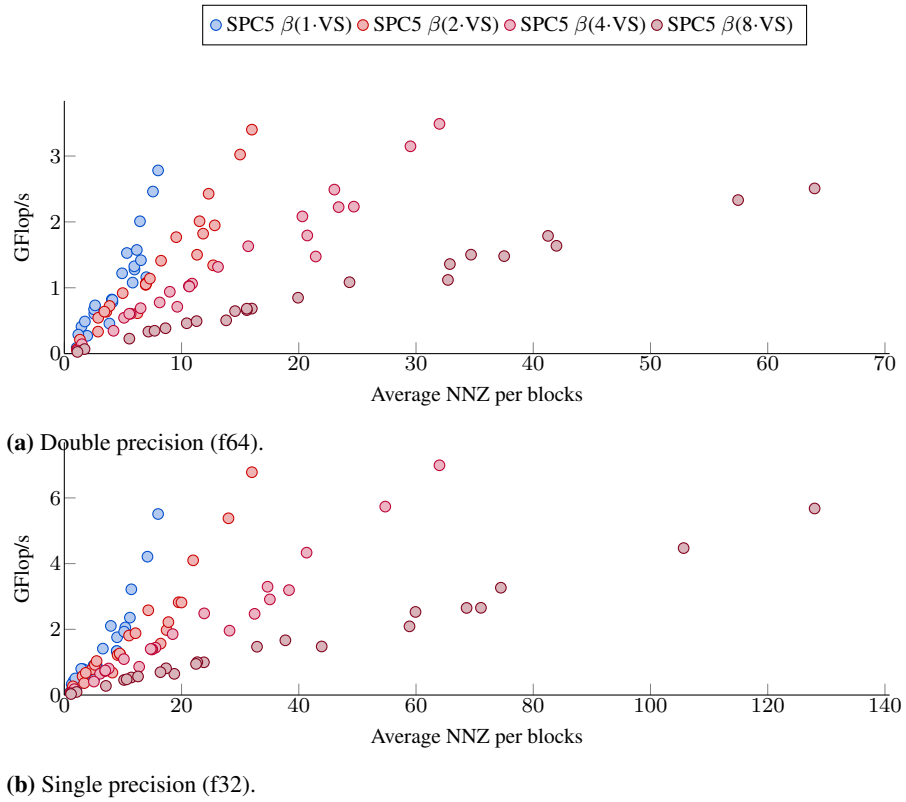


Fig. 4. Performance in Giga Flop per second for sequential computation in double and single precision for our SPC5 kernels on Fujitsu-SVE architecture for all the matrices of the test set

structions to reduce several SIMD vectors at the same time. This allows us to obtain a SIMD vector as output, which can then be used to update y with vectorized instructions. In the end, the performance difference between the two approaches is limited. However, for the rest of the study, we consider that the best Intel-AVX512 configuration is to use manual multi-reduction.

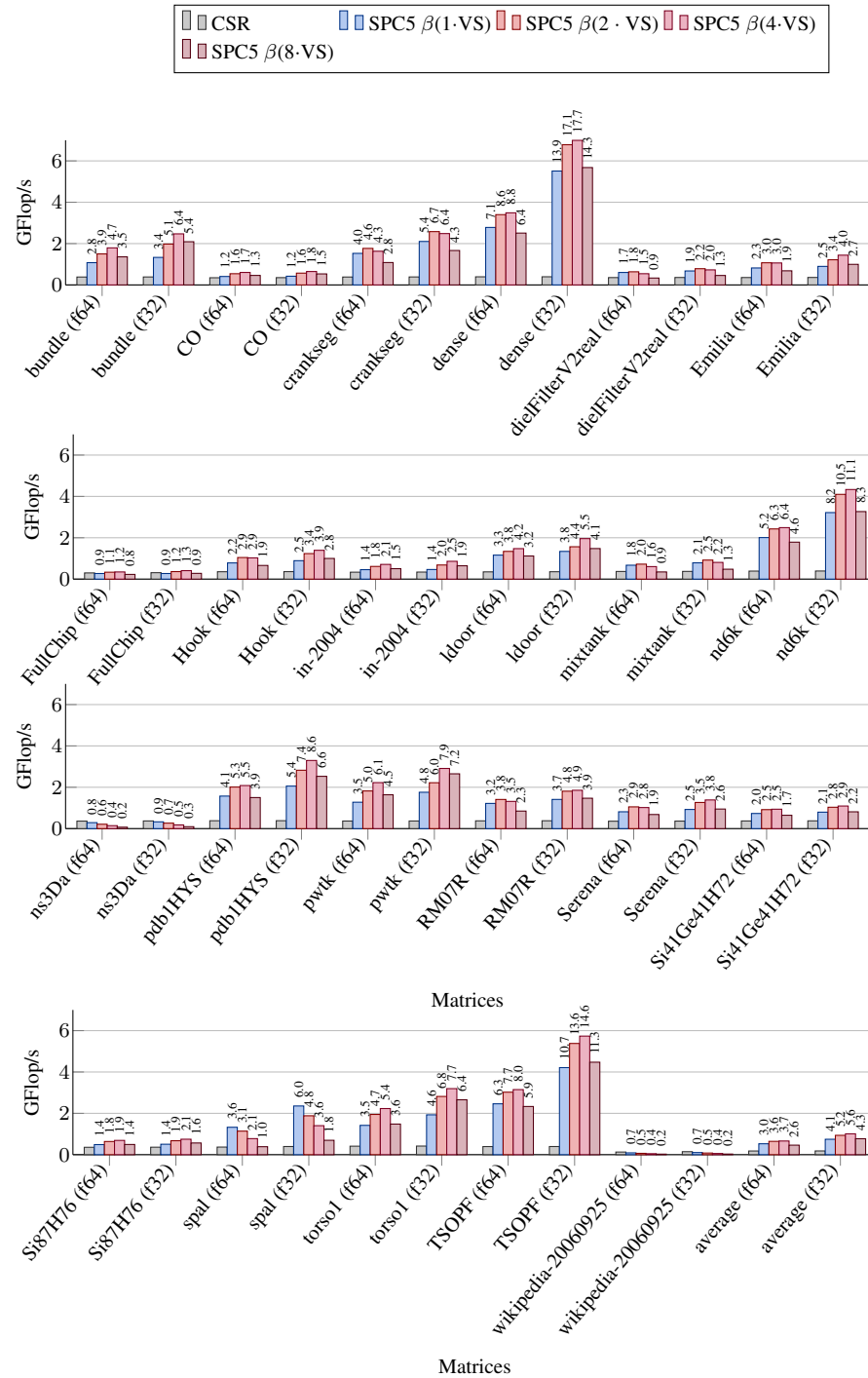


Fig. 5. Performance in Giga Flop per second for sequential computation in double and single precision for our SPC5 kernels on Fujitsu-SVE architecture. Speedup of SPC5 is computed against the scalar sequential version and written above the bars

	x load / reduction	CSR		$\beta(1,VS)$		$\beta(2,VS)$		$\beta(4,VS)$		$\beta(8,VS)$	
		f64	f32	f64	f32	f64	f32	f64	f32	f64	f32
CO	Yes/Yes	0.3	0.4	0.4 [x1.2]	0.4 [x1.2]	0.5 [x1.6]	0.6 [x1.6]	0.6 [x1.7]	0.6 [x1.8]	0.5 [x1.3]	0.5 [x1.5]
	Yes/No										0.5 [x1.4]
	No/Yes					0.5 [x1.5]		0.5 [x1.5]	0.6 [x1.6]	0.5 [x1.5]	0.6 [x1.6]
	No/No					0.5 [x1.5]		0.5 [x1.5]	0.6 [x1.6]	0.5 [x1.5]	0.6 [x1.6]
dense	Yes/Yes	0.4	0.4	2.8 [x7.1]	5.5 [x13.9]	3.4 [x8.6]	6.8 [x17.1]	3.5 [x8.8]	7.0 [x17.7]	2.5 [x6.4]	5.7 [x14.3]
	Yes/No						6.9 [x17.5]		6.5 [x16.4]	2.5 [x6.3]	6.4 [x16.0]
	No/Yes			2.8 [x7.0]		3.3 [x8.5]	6.8 [x17.2]	3.0 [x7.7]	6.4 [x16.1]	3.1 [x7.9]	6.4 [x16.1]
	No/No			2.8 [x7.0]	5.5 [x13.8]	3.3 [x8.5]	6.8 [x17.2]	3.0 [x7.7]	6.4 [x16.2]	3.1 [x7.9]	6.4 [x16.1]
nd6k	Yes/Yes	0.4	0.4	2.0 [x5.2]	3.2 [x8.2]	2.4 [x6.3]	4.1 [x10.5]	2.5 [x6.4]	4.3 [x11.1]	1.8 [x4.6]	3.3 [x8.3]
	Yes/No										
	No/Yes							2.2 [x5.8]	3.9 [x9.9]	2.0 [x5.2]	3.6 [x9.3]
	No/No							2.2 [x5.8]	3.9 [x9.9]	2.0 [x5.2]	3.6 [x9.3]
average	Yes/Yes	0.2	0.2	0.5 [x3.0]	0.7 [x4.1]	0.6 [x3.6]	0.9 [x5.2]	0.7 [x3.7]	1.0 [x5.6]	0.5 [x2.6]	0.8 [x4.3]
	Yes/No								1.0 [x5.5]		
	No/Yes					0.6 [x3.5]	0.9 [x5.0]	0.6 [x3.3]	0.9 [x4.8]	0.5 [x2.9]	0.8 [x4.6]
	No/No					0.6 [x3.5]	0.9 [x5.0]	0.6 [x3.3]	0.9 [x4.8]	0.5 [x2.9]	0.8 [x4.6]

(a) Fujitsu-SVE

	x load / reduction	CSR		MKL		$\beta(1,VS)$		$\beta(2,VS)$		$\beta(4,VS)$		$\beta(8,VS)$	
		f64	f32	f64	f32	f64	f32	f64	f32	f64	f32	f64	f32
CO	No/Yes	1.4	1.9	1.9 [x1.3]	2.3 [x1.2]	1.3 [x0.9]	1.4 [x0.8]	1.6 [x1.1]	1.9 [x1.0]	1.7 [x1.2]	1.9 [x1.0]	1.6 [x1.1]	1.9 [x1.0]
	No/No												
dense	No/Yes	1.2	1.3	2.3 [x1.9]	3.6 [x2.8]	3.7 [x3.0]	8.3 [x6.4]	4.1 [x3.4]	9.5 [x7.3]	4.3 [x3.6]	11.2 [x8.6]	4.4 [x3.6]	10.8 [x8.3]
	No/No						9.4 [x7.2]		10.6 [x8.1]	4.2 [x3.4]	11.0 [x8.5]		11.0 [x8.5]
nd6k	No/Yes	1.2	1.4	2.2 [x1.8]	2.8 [x2.0]	2.9 [x2.4]	6.2 [x4.5]	3.4 [x2.8]	7.3 [x5.4]	3.4 [x2.8]	7.4 [x5.4]	3.4 [x2.8]	7.4 [x5.4]
	No/No				2.8 [x2.1]		6.3 [x4.6]		7.3 [x5.3]		7.6 [x5.6]		7.1 [x5.2]
average	No/Yes	0.6	0.8	0.9 [x1.5]	1.2 [x1.6]	1.0 [x1.6]	1.7 [x2.3]	1.2 [x1.8]	2.0 [x2.6]	1.2 [x1.8]	2.0 [x2.7]	1.1 [x1.7]	1.9 [x2.5]
	No/No					1.1 [x1.7]	1.8 [x2.4]				2.0 [x2.6]		1.8 [x2.4]

(b) Intel-AVX512

Table 2. Performance in Giga Flop per second for sequential computation in double and single precision for our SPC5 kernels on Fujitsu-SVE and Intel-AVX512 architectures. Speedup of SPC5 is computed against the scalar sequential version, we print the values only when there is a difference with the first version (above one digit difference in the speedup). We provide the results for the CO, dense and nd6k matrices, and the average based on all the matrices from the test set. We compare the the loading of full x vectors per block (SVE and AVX512), and the use of manual multi-reduction against vectorial reduction (SVE only). The scalar and $\beta(1,VS)$ and MKL versions are expected to remain unchanged, differences for these kernels are from noise.

Best Configuration Detailed Results. We provide the complete results for Fujitsu-SVE in Figures 5 and 4. The results for Intel-AVX are shown in Figures 7 and 6.

In Figure 5, we can see that the performance of the SPC5 kernels is clearly related to the block filling. The performance model can be described as a constant cost per block that seems independent of the number of blocks or the number of NNZ. This means that the performance can be easily predicted from the block filling.

We also note that the performance increases as we increase the size of the blocks up to $4 \times VS$, but then it decreases for $8 \times VS$. This is more visible in Figure 5, where $\beta(8, VS)$ is the slowest SPC5 kernel in most cases.

The behaviors in single and double precision are similar. For some matrices, such as ns3Da, SPC5 is even slower than a simple CSR implementation. This means that the overhead of using vectorial instructions outweighs the benefits of vectorization since the block filling is very low.

The computation on the matrix TSOPF, which has a very high block filling, achieves performance almost equivalent to the dense matrix case. Finally, we can see the average performance in Figure 5 (last bars). While the speedup against CSR is significant, the raw performance is low compared to the peak performance of the machine.

The results for Intel-AVX are slightly different. In Figure 7, we can see that while there is a correlation between the block filling and the performance, the relationship is less clear than for Fujitsu-SVE.

We also note that the performance increases with the block size, such that the best performance is achieved with $\beta(8, VS)$. This is even more visible in Figure 7.

Contrary to Fujitsu-SVE, the performance obtained for TSOPF is not close to the dense matrix case. This means that while the blocks are almost full, the fact that we have to jump over the vector x has a negative impact on the performance.

The performance of SPC5 on Intel-AVX is higher than those obtained with Fujitsu-SVE for almost all matrices. Finally, SPC5 is faster than the Intel MKL CSR kernel for most matrices, but can be slower if there are less than two values per block.

Parallel Performance Overview. In Figure 8, we provide the results for the parallel executions. For the Fujitsu-SVE hardware, Figure 8a, for some matrices the speedup is above 42 (the number of cores). This is possible because the matrices are split and allocated by the threads such that each thread has its data on the memory nodes that correspond to its CPU core. In addition, the split of the matrices and the use of all the cores can result in using the cache more efficiently.

For the Intel-AVX512 hardware, Figure 8b, the executions on the dense matrix have poor performance for small blocks. This is clear that the x vector will be fully loaded for each row, such that the cache performance is tied to the final execution performance. We can notice that the speedup around 15 is far from the number of CPU cores (36). The workload balance between the threads is similar to the Fujitsu-SVE configuration, therefore, we consider that the difference comes from the memory organization and use.

5. Conclusion

We have presented a new version of our SPC5 framework, which remains efficient on architectures with AVX512 and is now compatible with ARM architectures with SVE. The

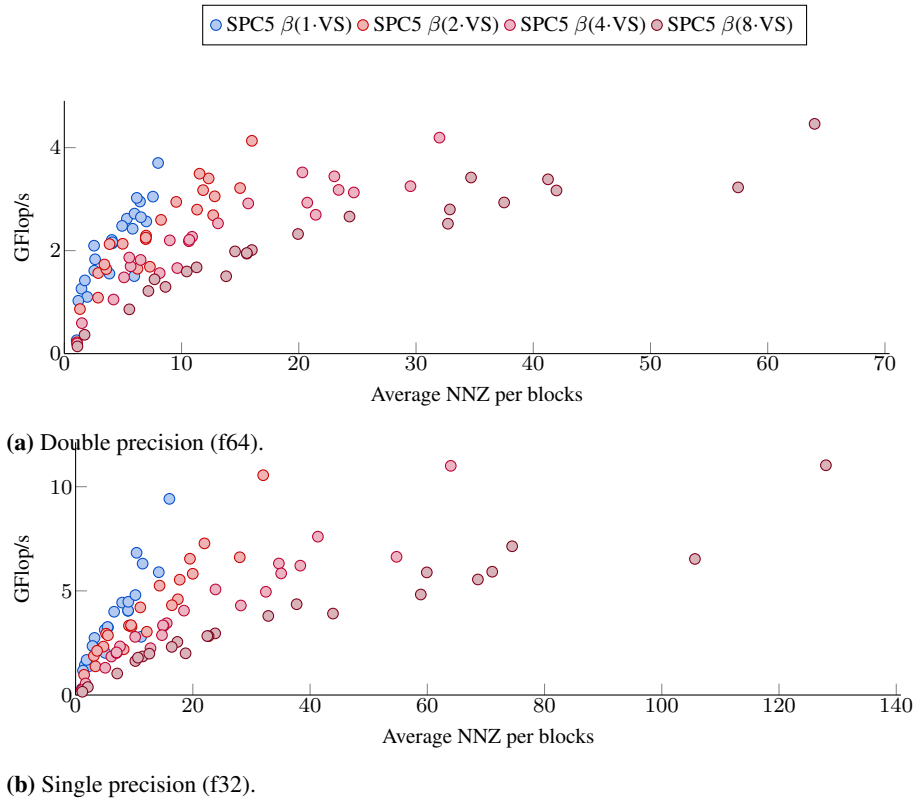


Fig. 6. Performance in Giga Flop per second for sequential computation in double and single precision for our SPC5 kernels on Intel-AVX architecture for all the matrices of the test set

same sparse matrix format can be used to target both ISA, allowing for interoperability and the use of a single framework on x86 and ARM-based architectures.

The SPC5's SpMV kernels are implemented differently, as they rely on an expansion mechanism of the NNZ (AVX512) or a compaction of the x vector (SVE). The performances we obtained are usually higher than a simple CSR kernel if there is more than a single NNZ per block. The $\beta(1,*)$ format has a low conversion cost as it leaves the array of NNZ unchanged compared to CSR, which makes it easy to plug in existing CSR-based applications.

In a future work, we would like to investigate if we could use a hybrid format, i.e., a format where we could have blocks of different sizes including blocks of scalar, to avoid using vectorial instructions when there is no benefit.

Acknowledgments. This work used the Isambard 2 UK National Tier-2 HPC Service⁶ operated by GW4 and the UK Met Office, which is an EPSRC project (EP/T022078/1). We also used the

⁶ <http://gw4.ac.uk/isambard/>

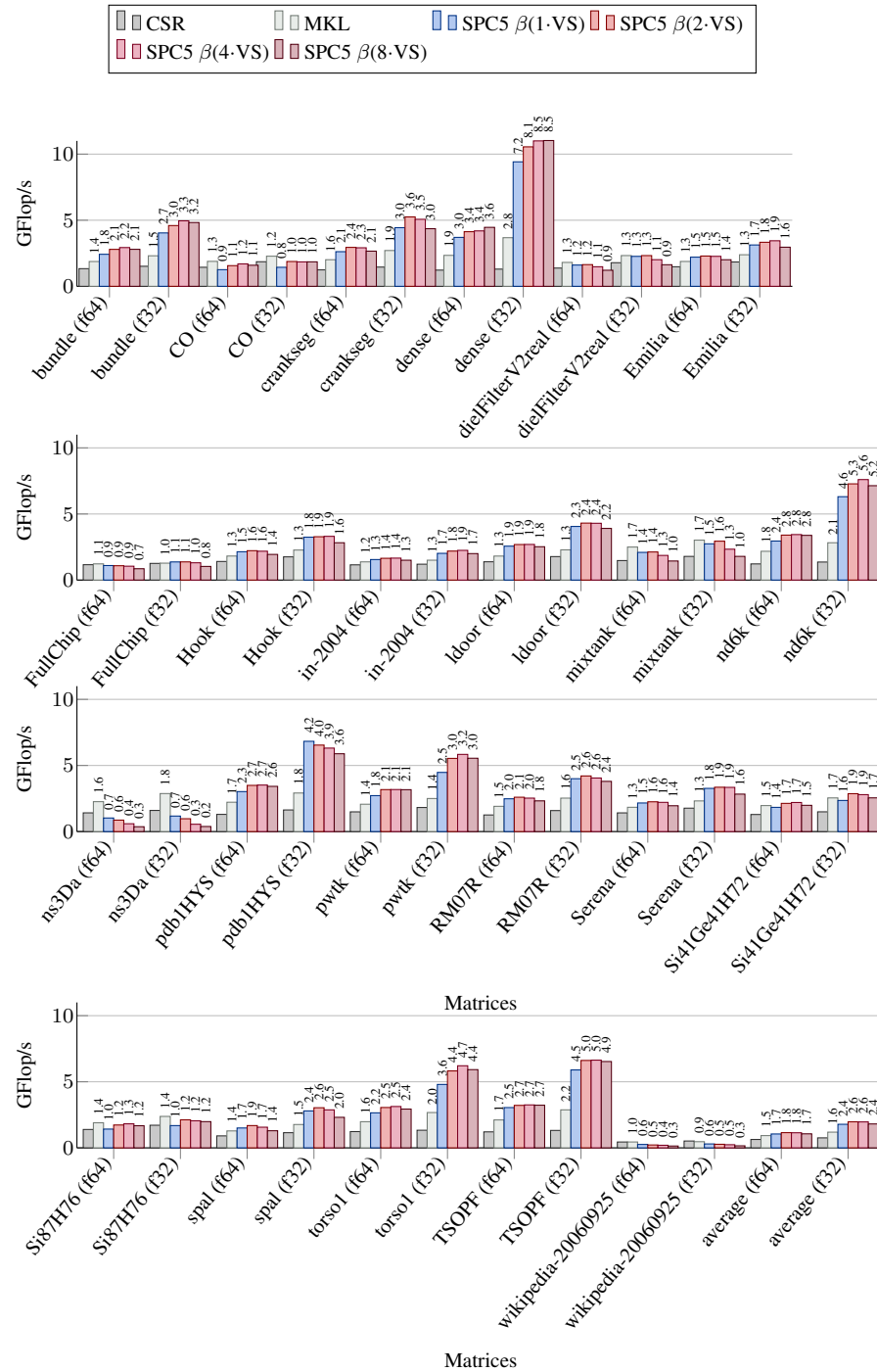


Fig. 7. Performance in Giga Flop per second for sequential computation in double and single precision for our SPC5 kernels on Intel-AVX512 architecture. Speedup of SPC5 is computed against the scalar sequential version and written above the bars

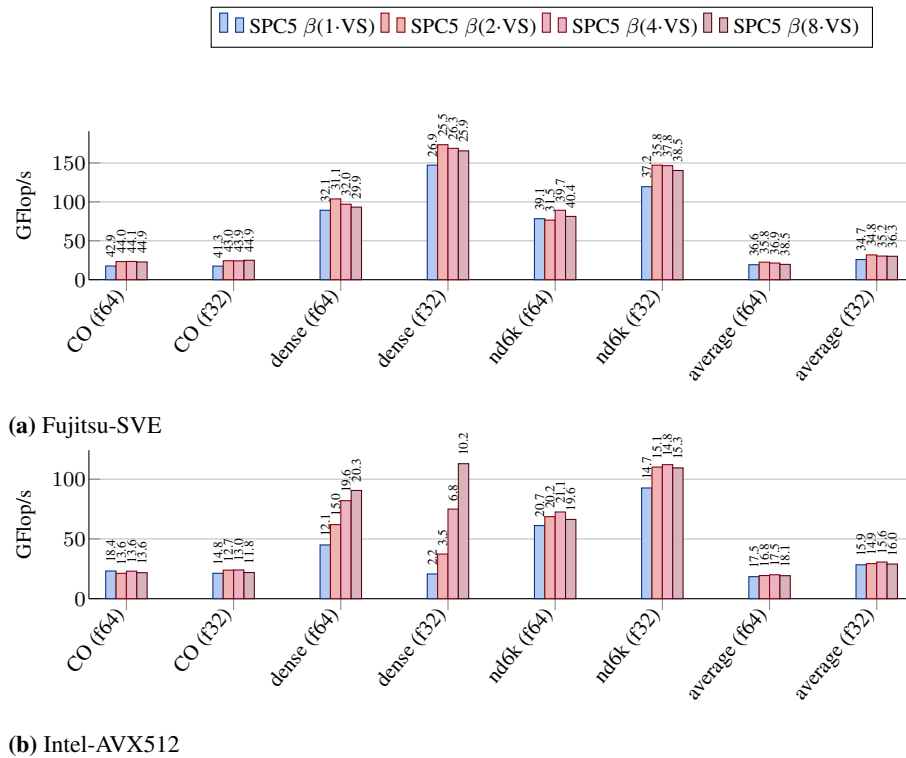


Fig. 8. Performance in Giga Flop per second for parallel computation in double and single precision for our SPC5 kernels on Fujitsu-SVE and Intel-AVX512 architectures. Speedup of parallel SPC5 is computed against the same sequential version and written above the bars. We provide the results for the CO, dense and nd6k matrices, and the average based on all the matrices from the test set.

PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Universite de Bordeaux, Bordeaux INP and Conseil Regional d'Aquitaine ⁷. In addition, this work used the FarmSVE library [6].

References

- Alappat, C., Meyer, N., Laukemann, J., Gruber, T., Hager, G., Wellein, G., Wettig, T.: Ecm modeling and performance tuning of spmv and lattice qcd on a64fx. arXiv preprint arXiv:2103.03013 (2021)
- AOKI, R., MURAO, H.: Optimization of x265 encoder using arm sve
- ARM: Arm architecture reference manual supplement, the scalable vector extension (sve), for armv8-a. <https://developer.arm.com/documentation/ddi0584/ag/>, accessed: July 2020 (version Beta)

⁷ <https://www.plafrim.fr>

4. ARM: Arm c language extensions for sve. <https://developer.arm.com/documentation/100987/0000>, accessed: July 2020 (version 00bet1)
5. Bramas, B.: Optimization and parallelization of the boundary element method for the wave equation in time domain. Ph.D. thesis, Université de Bordeaux (2016)
6. Bramas, B.: Farm-SVE: A scalar C++ implementation of the ARM® Scalable Vector Extension (SVE) (Jul 2020), <https://inria.hal.science/hal-02906179>
7. Bramas, B., Kus, P.: Computing the sparse matrix vector product using block-based kernels without zero padding on processors with avx-512 instructions. *PeerJ Computer Science* 4, e151 (Apr 2018), <https://doi.org/10.7717/peerj-cs.151>
8. Cuthill, E., McKee, J.: Reducing the bandwidth of sparse symmetric matrices. In: *Proceedings of the 1969 24th national conference*. pp. 157–172. ACM (1969)
9. Davis, T.A., Hu, Y.: The university of florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)* 38(1), 1 (2011)
10. Domke, J.: A64fx—your compiler you must decide! arXiv preprint arXiv:2107.07157 (2021)
11. Flynn, M.J.: Very high-speed computing systems. *Proceedings of the IEEE* 54(12), 1901–1909 (1966)
12. Fujitsu: A64fx microarchitecture manual. https://github.com/fujitsu/A64FX/blob/master/doc/A64FX_Microarchitecture_Manual_en_1.8.1.pdf (2022), accessed on 26 July 2023 from <https://www.fujitsu.com/global/products/computing/servers/supercomputer/a64fx/>
13. Im, E.J., Yelick, K., Vuduc, R.: Sparsity: Optimization framework for sparse matrix kernels. *International Journal of High Performance Computing Applications* 18(1), 135–158 (2004)
14. Kodama, Y., Odajima, T., Matsuda, M., Tsuji, M., Lee, J., Sato, M.: Preliminary performance evaluation of application kernels using arm sve with multiple vector lengths. In: *2017 IEEE International Conference on Cluster Computing (CLUSTER)*. pp. 677–684 (2017)
15. Kreuzer, M., Hager, G., Wellein, G., Fehske, H., Bishop, A.R.: A unified sparse matrix data format for efficient general sparse matrix-vector multiplication on modern processors with wide simd units. *SIAM Journal on Scientific Computing* 36(5), C401–C423 (2014)
16. Liu, W., Vinter, B.: Csr5: An efficient storage format for cross-platform sparse matrix-vector multiplication. In: *Proceedings of the 29th ACM on International Conference on Supercomputing*. pp. 339–350. ACM (2015)
17. Liu, X., Smelyanskiy, M., Chow, E., Dubey, P.: Efficient sparse matrix-vector multiplication on x86-based many-core processors. In: *Proceedings of the 27th international ACM conference on International conference on supercomputing*. pp. 273–282. ACM (2013)
18. Meyer, N., Georg, P., Pleiter, D., Solbrig, S., Wettig, T.: Sve-enabling lattice qcd codes. In: *2018 IEEE International Conference on Cluster Computing (CLUSTER)*. pp. 623–628 (2018)
19. Pichel, J.C., Heras, D.B., Cabaleiro, J.C., Rivera, F.F.: Performance optimization of irregular codes based on the combination of reordering and blocking techniques. *Parallel Computing* 31(8), 858–876 (2005)
20. Pinar, A., Heath, M.T.: Improving performance of sparse matrix-vector multiplication. In: *Proceedings of the 1999 ACM/IEEE conference on Supercomputing*. p. 30. ACM (1999)
21. Stephens, N., Biles, S., Boettcher, M., Eapen, J., Eyole, M., Gabrielli, G., Horsnell, M., Maglis, G., Martinez, A., Premillieu, N., Reid, A., Rico, A., Walker, P.: The arm scalable vector extension. *IEEE Micro* 37(2), 26–39 (Mar 2017), <https://doi.org/10.1109/MM.2017.35>
22. Varela, M.H.: Manycore Architectures and SIMD Optimizations for High Performance Computing. Ph.D. thesis, Universidade da Coruña (2022)
23. Vuduc, R.W., Moon, H.J.: Fast sparse matrix-vector multiplication by exploiting variable block structure. In: *High Performance Computing and Communications*, pp. 807–816. Springer (2005)
24. Vuduc, R.W.: Automatic performance tuning of sparse matrix kernels. Ph.D. thesis, Citeseer (2003)

25. Wan, X., Gu, N., Su, J.: Accelerating level 2 blas based on arm sve. In: 2021 4th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEM-CSE). pp. 1018–1022 (2021)

Evann Regnault holds a Bachelor of Science degree and is currently pursuing a Master of Science in Computer Science at the University of Strasbourg. His research focuses on low-level optimizations and software engineering for High-Performance Computing (HPC).

Bérenger Bramas is a permanent research scientist at Inria Nancy and the ICube laboratory. He possesses dual Master of Science degrees: one in Computer Science from the University Blaise Pascal and the other in Software Engineering from ISIMA. He earned his Ph.D. from Bordeaux University. His research focuses on High-Performance Computing (HPC), including optimizations, complex algorithm development, accelerators, and software engineering for HPC. Additionally, his work involves scientific computing, runtime systems, scheduling, and the development of tools for automatic vectorization and parallelization.

Received: August 19, 2023; Accepted: December 12, 2023.

Evaluation of Deep Learning Techniques for Plant Disease Detection^{*}

Cedric Marco-Detchart¹, Jaime Andrés Rincon², Carlos Carrascosa¹, and Vicente Julian^{1,3}

¹ Valencian Research Institute for Artificial Intelligence (VRAIN),
Universitat Politècnica de València (UPV),

Camino de Vera s/n, 46022, Valencia, Spain {cedmarde, carrasco, vjulian}@upv.es

² Departamento de Digitalización, Escuela Politécnica Superior,
Universidad de Burgos, 09006, Burgos, Spain
jarincon@ubu.es

³ Valencian Graduate School and Research Network of Artificial Intelligence (VALGRAI),
Universitat Politècnica de València (UPV),
Camí de Vera s/n, 46022 Valencia, Spain

Abstract. In recent years, several proposals have been based on Artificial Intelligence techniques for automatically detecting the presence of pests and diseases in crops from images usually taken with a camera. By training with pictures of affected crops and healthy crops, artificial intelligence techniques learn to distinguish one from the other. Furthermore, in the long term, it is intended that the tools developed from such approaches will allow the automation and increased frequency of plant analysis, thus increasing the possibility of determining and predicting crop health and potential biotic risks. However, the great diversity of proposed solutions leads us to the need to study them, present possible situations for their improvement, such as image preprocessing, and analyse the robustness of the proposals examined against more realistic pictures than those existing in the datasets typically used. Taking all this into account, this paper embarks on a comprehensive exploration of various AI techniques leveraging leaf images for the autonomous detection of plant diseases. By fostering a deeper understanding of the strengths and limitations of these methodologies, this research contributes to the vanguard of agricultural disease detection, propelling innovation, and fostering the maturation of AI-driven solutions in this critical domain.

Keywords: Plant Disease Detection, Classification, Image Preprocessing, Deep Learning.

1. Introduction

The growing interest of the agri-food sector in the new solutions that digitalisation can provide, and the support given to their development by public administrations and different organisations, is facilitating an increasing number of technological initiatives at the international and national levels.

One factor influencing crop yields is the possible incidence of pests and diseases. To reduce their impact, farmers make recurrent use of phytosanitary products. In this way,

^{*} This paper is an extended version of an ASSIA workshop paper.

the development of potentially dangerous populations can be controlled, thus ensuring production. This group also includes herbicides, which prevent competition for nutrients, water, and the establishment of the main crops with other unwanted plants.

In many cases, given the uncertainty of the time of pest emergence and its virulence, farmers often carry out preventive treatments. Over the last few years, the cost of these treatments has shown a clear upward trend. For example, in Spain, the average price per hectare has gone from 50 euros in 2009 to 88 euros in 2019⁴. In other words, in just 11 years, this component has undergone an increase of more than 76%.

Within the same EU strategy to reduce the environmental impact of European agriculture, another objective has been set to reduce the use of phytosanitary products by 50% and make more rational use of these products from both an environmental and economic point of view. To achieve this, digitalisation is presented as an essential tool. In this sense, disease detection and pest evolution models can be created so that treatments are only carried out when they pose a risk to production.

According to this, plant disease is nowadays a significant concern in agriculture. There are many involved counterparts, from economic to climatic and social consequences. Early disease detection and control is one of the cornerstones of preventing economic waste and production losses. Plant disease detection is a complex task and has been traditionally done by ocular inspection and through the personal experience of farmers. One of the main concerns is to be as quick as possible to detect the disease before it spreads through the land. Furthermore, the plant is not necessarily affected by a single disease; multiple conditions can show up simultaneously, increasing the detection difficulty. Hence, the wide variety of plants combined with the different diseases nowadays makes experts fail in their tasks. With the significant increase of digital imagery, there is an opportunity to generalise the expert's knowledge and use image processing for automatic plant disease detection.

Soil analysis is traditionally done using satellite and hyperspectral images, which permits the study of wide land extensions and characteristics that humans cannot see. The counterpart of this type of analysis is that the quantity of available images is limited, and that concrete diseases can only be seen at the field level. As for hyperspectral images, the equipment needed is expensive. To make the disease detector system accessible, it must be low-cost so anyone can use it. It should work with images in the visible range, that is, in RGB space.

The current situation in the agricultural sector exacerbates the problems of lack of experts and, therefore, of time available to detect plant diseases. In this sense, the automatic detection of plant diseases plays a crucial role in overcoming these problems and providing an alternative solution to the damage caused by plant diseases.

Automatic identification of plant diseases presents several challenges ranging from problems in the capture process, such as noise or fog over the camera, to unwanted information in the images themselves, such as an inappropriate background, the soil itself, or other plants that interfere with the identification.

Deep learning-based solutions are considered the most suitable for this type of problem. Deep Learning techniques, particularly Convolutional Neural Networks, rely on weight optimisations, searching for maxima in the parameter space. In terms of statistical results, *e.g.* in edge detection, they have achieved exceptional results, even surpassing hu-

⁴ <https://www.mapa.gob.es/es/estadistica/temas/publicaciones/anuario-de-estadistica/>

man performance [30]. As a counterpart, due to their nature, neural networks can undergo with difficulties to adapt to real scenarios when the tested data is considerably different from the training data [28]. Even so, there are still aspects to be taken into account since the proposed models are very dependent on the characteristics of the dataset used. Alternative datasets more adjusted to real situations can easily alter the accuracy of deep learning-based solutions.

This work reviews the most relevant network architectures used in the literature to detect plant disease. The experiments are conducted to classify individual diseases in plant images. Moreover, we analyse and compare their performance and investigate whether preprocessing the images before the training step impacts the correct classification. As it is difficult to say whether preprocessing has a real impact, we evaluate the different image set variants over a real image synthetic dataset, analysing the robustness of each network configuration.

The paper is structured as follows. In Section 2 we review related work in the automatic detection of plant disease. Then, Section 4 presents the use of preprocessing as the first step before training image classification networks along with the network families used. Following this, Section 5 shows the experimental framework devised. Finally, a discussion is presented in Section 5.1 along with some conclusions about the results obtained.

2. Related Work

Automatic plant disease detection is a classification problem done through the analysis of image features, which can be as various as geometric or colour features. In addition, specific indexes such as the NDVI-index, which measure the level of green on images, are commonly used (but in hyperspectral images). As a similar index type, but for visible range images, alternatives are used, such as the VARI index or the vNDVI index [7].

As indicated by Barbedo [3] and Hasan [16], there are a series of challenges when identifying plant diseases and managing crops automatically (as seen in Figure 1). One of the main problems with focussing the interest on the plant or leaf itself is the background removal, which can be soil or other plants. In addition, capture level problems can be an issue as brightness or occlusions. In terms of the disease itself, there might not be a unique disease or different diseases can have similar characteristics or might not be ideally defined.

Some of these challenges might be tackled through a pre-processing step, where an image is treated, so that spurious information is removed, *e.g.* background segmentation or texture removal (smoothing [25, 29]) or even image improvement (*e.g.* contrast enhancement [23]).

Once the images have been processed, they can be classified by two types of techniques. On the one hand, Machine Learning (ML) techniques aim at classifying plant diseases based on different features extracted from images (geometric characteristics, colour information, gradients, etc.). Different classifiers such as Support Vector Machines [6,31], K-means classifier [13], and Random Forest [20] are used to detect different plant diseases. These techniques need a very precise human-made solution (ground truth) and assistance to be performant. They work well when there is a limited amount of data. On the other hand, despite their lack of explainability, Deep Learning (DL) approaches have been

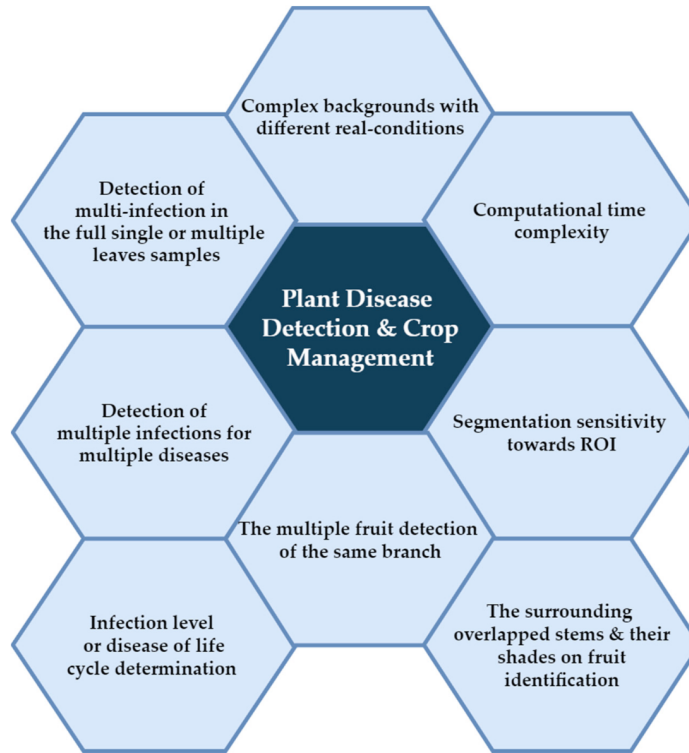


Fig. 1. Challenges in plant disease detection and crop management [16]

extensively used nowadays and report promising results. In the literature, some recurrent examples of well-known Convolutional Neural Networks (CNNs) are used to detect plant diseases. As is well known, even if there are increasingly more available images to work with, the quantity/quality is limited to learning in a specific task. In those cases, Transfer Learning is used to build a network based on pre-trained information and adapted to the concerned task. These networks are pre-trained on large datasets *e.g.* ImageNet [8]. This process takes the first layer of the trained network and removes the last layers adapting them to the specific task and training only these last steps. In this way, the specific task is not trained from scratch, and the computing time is shortened. The most common and efficient networks in the literature are *Alexnet* [21], *ResNet50* [18], *VGG16* [36], *Inception V3* [37]. *EfficientNet* [38] can also be considered a group of networks, as there are 8 types of subnets.

We can see in [14] that their model, based on *ResNet50* and pre-trained on Imagenet, is robust against field information (soil, background, etc...), obtaining an accuracy of 98.5% with synthetic soil background and 72.03% on real field images. They work with data augmentation and real field images from plants to do so. In addition, as the same disease on different plants is considered a different class, the authors propose to use a species-specific

classification as the disease symptoms can be pretty similar. In this way, geometric, colour, and texture features are learnt and then incorporated into the baseline model.

Atila *et al.* propose in [2] the use of EfficientNet in its different variants (B0 to B7). The particularity of this neural network is that, as a first step, it determines the best scaling dimensions and it does not use the well-known Rectifier Linear Unit (ReLU) as an activation function, using the Swish function instead. In this work, the model is pre-trained with ImageNet and adapted by removing the Fully Connected layers with 1000 outputs to fit the 39 possible outputs of the PlantVillage dataset. The results obtained with their proposal get an accuracy beyond 99.6% compared to state-of-the-art neural nets, all pre-trained with ImageNet.

An interesting approach is that of Schwarz Schuler *et al.* [35] which propose an InceptionV3 based architecture where the image processing is separated into two branches where each image channel, in Lab colour space, is treated separately and then fused to better adapt to the inherent characteristics of each channel. The authors test their approach by varying the importance of each channel with the percentage of filters used for each of the branches. The performance of the process in all cases is greater than 99% of accuracy.

An alternative approach to mostly used network architecture is that of Capsule Networks [32] which fixes one of the significant drawbacks of a standard CNN. CNNs do not consider the possible feature hierarchy in an image considering similar images as equal even when they are not. In the work presented by Samin *et al.* [33] the Capsule Network approach is used without using Transfer Learning, obtaining an accuracy of 93.07%.

More recently, a lightweight CNN approach was based on Inception and Residual connection [17]. The proposed approach extracts better features from the input images. This is done by replacing the standard convolution by a depth-wise separable convolution combined with a point-wise convolution, which results in fewer parameters as well as a speed-up in the processing. The resulting performance with the approach presented is 99.39% accuracy.

After studying the state-of-the-art, it can be seen that there are currently a multitude of proposals, most of them based on deep learning techniques that offer promising results from the existing datasets. However, there are specific gaps that we think should be analysed. On the one hand, some works suggest the need for image pre-processing before classification; in our opinion, this aspect should be studied in greater detail as it may allow for an improvement in the classification process. On the other hand, most of the works are evaluated against a so-called ideal dataset. Using more realistic datasets to validate existing models would allow for analysis of their possible robustness. Nevertheless, [5] is an interesting approach, but as they are working with infrared images, they would need to use infrared cameras when applied to the real world, which are expensive to deploy.

In summary, the reviewed works exhibit several shortcomings. First, there is a common challenge in background removal, which includes soil or other plants. This issue can be addressed through pre-processing steps, such as background segmentation and texture removal. Additionally, the variability in disease symptoms and the lack of a unique or well-defined disease pose challenges. To tackle this, the proposed work employs a species-specific classification approach, which considers the similarity of symptoms among diseases within a species. Moreover, most of the existing works rely on ideal datasets for evaluation, which may not reflect real-world conditions. To address this, the proposed work emphasizes the need for more realistic datasets to validate models and analyze their

robustness. Additionally, some works suggest image pre-processing before classification, and the proposed work aims to delve deeper into this aspect to potentially improve the classification process. Finally, while the reviewed works primarily focus on deep learning techniques, the proposed work offers a comprehensive comparative study of various approaches, including deep learning and alternative network architectures. This comparative analysis helps identify the strengths and weaknesses of different methods and contributes to a better understanding of plant disease detection techniques. In the following section, the proposed study is presented.

3. Contextualizing Plant Disease Detection

This section provides a comprehensive overview of the problem statement, research objectives, and motivation driving our study on evaluating a collection of different neural network architectures for plant disease detection.

3.1. Problem Statement

The accurate classification of plant diseases is a matter of utmost importance in contemporary agriculture. With the global population rising and increasing pressure on sustainable food production, the need for precise disease detection has never been greater.

Early and precise disease identification facilitates timely intervention, curbing the spread of infections and minimising crop losses. This ensures a stable food supply to meet the demands of growing populations. Accurate classification ensures that infected productions are promptly identified and removed, safeguarding consumer health and maintaining the reputation of agricultural products. Reducing chemical pesticide use and optimising resource allocation in farming can be achieved through targeted disease management, contributing significantly to sustainable agricultural practices.

However, several challenges and implications loom over the issue of plant disease classification. Misclassification can lead to unnecessary treatments or neglect of infected plants, resulting in economic losses for farmers. In addition, it can lead to environmental pollution and harm to non-target species and compromise food security by reducing crop yields and impacting the availability and affordability of agricultural products. Accurate datasets for plant disease classification are essential for advancing research in crop protection and breeding resilient plant varieties.

3.2. Objectives and motivation

Our proposal investigates various deep learning models' effectiveness in the automated detection and classification of plant diseases, rigorously evaluating their performance. Based on the performance measures obtained, we determine which neural network architectures and transfer learning strategies yield the best results for plant disease detection. With the proposed experiments, we bridge the gap between research and practical application by developing a user-friendly tool that can assist farmers in identifying plant diseases quickly and accurately.

Our motivation to embark on this study is grounded in the pressing need to address the challenges posed by plant diseases in agriculture. We aspire to provide a practical and efficient solution to the longstanding problem of plant disease detection, making a tangible

impact on the agricultural industry. Ensuring a stable and secure food supply for current and future generations is a driving force behind our work, guarding crop yields against the persistent threat of plant diseases, contributing to reducing environmental impact due to excessive pesticide use and promoting responsible farming practices. By exploring novel approaches for automated disease detection in plants, we seek to advance the fields of computer vision and machine learning, pushing the boundaries of technological innovation.

4. Preprocessing and Classification models

In this Section, we briefly present the preprocessing step that we use as a primary task to train the networks and build specific sets of images with regularisation and sharpening. We also present the different network architectures we put to the test and the different parameters we configure.

4.1. Preprocessing

As the first task in this work, we consider the preprocessing step inspired by the Bezdek Breakdown Structure (BBS) [4] for edge detection. Four steps characterise the process: image conditioning, feature extraction, blending and scaling.

This structure helps to understand edge detection as four independent phases. For our purpose of interpreting plant leaf classification using Convolutional Neural Networks (CNN), in some way, we take the first step of the BBS as the input of our neural system. We do not consider the rest of the steps as they are edge-detection-specific.

Image conditioning focuses on removing noise, or additional information that is not needed for the process it is undergoing (in this case, feature extraction in the different layers of the network).

It is a cornerstone task as it removes spurious information on the image due to the capture process or reduces texture information. In our approach, we propose to use The Gravitational Smoothing (GS) process [25] as a conditioning step in the preprocessing. This preprocessing step is done to all the images of the dataset in order to obtain a variation that will be used for training purposes individually without mixing the new images with the original ones.

GS is a content-aware smoothing method that adapts the image regularisation based on local information, as opposed to Gaussian smoothing, whose main problem is the blurring of objects in the image. GS simulates the movement of pixels in a 5D spatial-tonal space, bringing closer or moving away similar/dissimilar pixels. This behaviour allows us to control the blurring effect by removing unwanted information such as textures while preserving abrupt tonal changes (*e.g.* significant colour variation). In this technique, each pixel is considered a particle in the space that exerts an attractive or repulsive force on every surrounding pixel. Then the total force over a pixel is the sum of all individual forces around a pixel. Each force is computed as the product of a gravitational constant G and the inverse of the distance in the spatial-tonal space. Additionally, this force model can be used oppositely, changing the direction of the forces. Using GS in this way enhances the image's characteristics, highlighting the differences, which in our case can be beneficial to detect the diseased regions better. An example of the resulting images obtained can be

seen in Figures 2(a)-2(b), where we apply the smoothing and sharpening version of GS to an original leaf image.

4.2. Deep Neuronal Network Description

Different network architectures were evaluated to perform this classification of plant diseases to determine which configuration allowed the best results to be obtained. Different aspects were taken into account, such as transfer learning, data augmentation and the end device: server or edge device. In order to obtain the different networks, a series of hyperparameters were modified to each of the networks. In order to determine which configuration allowed the best results to be obtained.

The hyperparameters presented below were the result of an extensive iterative process. These specific hyperparameters were identified as the ones yielding the most favorable outcomes during the validation phase. Notably, the decision to set the number of training epochs at 7 emerged from a careful consideration of model performance. Extensive experimentation revealed that increasing the number of epochs led to a phenomenon known as overtraining, where the model became overly specialized to the training data, compromising its generalization capability. Conversely, reducing the number of epochs resulted in markedly suboptimal results, indicating the need for a moderate yet effective training duration. This strategic choice of 7 epochs reflects a delicate balance, ensuring that the model captures underlying patterns without succumbing to overfitting, thus optimizing its predictive capacity.

- **Epochs:** 7
- **Network:** InceptionV3, MobilenetV2, NASNetMobile, Efficientnet-B0, EfficientnetV2-Imagenet1k, EfficientNet-Imagenet21k
- **Learning rate:** 0.001
- **Transfer learning (Tl):** Yes or No
- **Data augmentation (Da):** Yes or No
- **Dataset:** Raw, Mixed, Smoothed, Sharpened

Each of the different networks used has several characteristics that make them stand out from each other. Out of these characteristics we can highlight the ability to be used in systems with ARM architecture. Specifically, if it is possible to run the model efficiently on Edge devices, TPU (Tensor Processing Unit), CPU or GPU. The characteristics of each of the networks used are described below.

Inceptionv3 [37], [26] is an image recognition architecture that has been shown to achieve an accuracy of better than 78.1% on the ImageNet dataset. The Inceptionv3 model comprises symmetric and asymmetric building blocks, including convolutions, mean reduction, maximum clustering, concatenations, dropouts and fully connected layers. Batch normalisation is a widely used technique in Inception v3 and is applied to activation inputs. Inception v3 uses the Softmax function to perform the loss function calculation, which is ideal for non-binary classifications.

The **MobileNetV2** [34] network is an evolution of its predecessor MobileNetV1. These networks belong to a family of neural networks widely used in computer vision. They are general-purpose networks designed especially for use on mobile devices and work very well on devices such as a Raspberry Pi [12]. MobileNetV2 is a significant

improvement over MobileNetV1 and presents a breakthrough in visual recognition on low-power devices. It enables devices to perform classification, object detection and semantic segmentation. In the NASNetMobile [27] network, the authors propose to search for an architectural block using a small dataset, in order to then be transferred to a larger dataset. In the NASNet network, they first seek to obtain the best convolutional layer using the CIFAR-10 dataset. Once this layer is obtained, it is applied to the ImageNet dataset, stacking more copies of the obtained convolutional layer. At the same time, the authors of the NASNet network propose a new regularisation technique called ScheduledDropPath, which significantly improves generalisation in NASNet models. One of the advantages of the NASNet network is the reduction of the generated model size.

EfficientNet [38] is a convolutional neural network architecture and scaling method that uniformly scales all depth/width/resolution dimensions using a composite coefficient. Unlike conventional practice that arbitrarily scales these factors, the EfficientNet scaling method uniformly scales the network's width, depth and resolution with a set of fixed scaling coefficients.

EfficientNet-B0 [38] is one of the configurations of the **EfficientNet** family that aims to scale ConvNets, maintaining and even surpassing their efficiency. Scaling is uniformly performed by a given coefficient unlike other methods where it is done arbitrarily. The variant B0 is the basic one and is based on the bottleneck residual blocks from MobileNetV2. **EfficientnetV2-B0** [39] is an evolution of the previous family of neural nets that improves in terms of training times and parameter efficiency. In addition, the authors perform a reduction of a network that is almost 6.8 times smaller. In our experiments, we use two variants of EfficientNetV2-B0, one pre-trained over a subset of Imagenet (EfficientnetV2-Imagenet1k) and the other over the complete Imagenet (EfficientNet-Imagenet21k).

4.3. Discussion

The use of different types of neural network models makes it possible to determine which type of network has the best classification rate. However, the use of convolutional neural networks (CNNs) has been widely used in the classification of plant diseases from RGB images [22]. However, these models do not necessarily focus on the visible parts affected by a plant disease for classification, and can sometimes take into account irrelevant backgrounds or healthy parts of the plant.

However, each of the models you mentioned, InceptionV3, MobileNetV2, EfficientNet and EfficientNetV2-B0, has unique architectural features that may make them suitable for plant disease classification. Here is a brief discussion of each:

1. **InceptionV3:** This model is known for its inception modules, which allow for efficient computation and deep networks through a carefully crafted design. The inception modules help the network learn useful representations at multiple scales, which can be beneficial for plant disease classification where symptoms can vary in size [10].
2. **MobileNetV2:** This model introduces two new features to the architecture: Linear Bottlenecks and Inverted Residuals [11] [24]. The use of linear bottlenecks is to maintain the representational power. The inverted residuals allow for a reduction in

computational complexity. This makes MobileNetV2 a lightweight model that can be used on devices with limited computational resources.

3. **EfficientNet:** EfficientNet uses a new scaling method that uniformly scales all dimensions of depth/width/resolution using a simple yet highly effective compound coefficient. This could potentially capture more complex features in the images, making it suitable for plant disease classification [1].
4. **EfficientNetV2-B0:** This model improves upon EfficientNet by introducing a form of progressive learning mechanism which expands the network topology gradually over the course of training process improving the model's learning capacity. This could be particularly useful in plant disease classification where the dataset might be imbalanced [9].

The choice of a model may depend on several aspects, such as the size and quality of the dataset, the available computing resources and the specific needs of the task. For example, MobileNetV2 may be the best choice if you are working with limited computational resources due to its remarkable efficiency. On the other hand, if you are dealing with a large and intricate dataset, EfficientNet or EfficientNetV2-B0 might be more suitable options given their ability to handle complex features. The choice of a model may depend on several aspects, such as the size and quality of the dataset, the available computing resources and the specific needs of the task. For example, MobileNetV2 may be the best choice when working with limited computing resources due to its remarkable efficiency. On the other hand, when datasets are large and intricate, EfficientNet or EfficientNetV2-B0 may be more suitable options given their ability to handle complex features. On the other hand, it is important to generate instances that reflect real-world scenarios, all in order to improve the performance of the machine learning model. However, it is crucial to understand that not all data augmentation techniques are a one-size-fits-all solution for all datasets. In the experiments conducted, image transformations and rotations were applied, a technique that is usually reserved for image datasets.

5. Experimental Framework

In this Section, we compare the most usual proposals found in the literature for plant disease detection. In Section 5.1, we briefly introduce the dataset used for our experiments as well as the performance measures we used to quantify the results. In Section 5.2, we show the quantitative results obtained in the comparison along with its analysis.

5.1. Dataset and Quantification of the Results

For our experiments, we have put to the test our proposal with the PlanVillage dataset [19] which contains 54303 images of healthy and diseased leaf plants classified by species and disease, having a total of 38 classes.

In addition, the data set has three different types of images available. The main raw image is an RGB colour space image taken on a uniform background. In addition, there is a grey-scale version of the previous image. And finally, a segmented version, where the background has been removed and only contains the leaf information.

From the original dataset, we prepare four different derivatives:

- **RAW**: It consists of images of the original dataset with colour and background.
- **Mixed**: Contains a mixture of RAW images, rotated and with background removed.
- **Smoo**: This dataset contains images of RAW where the background has been removed, and the leaf has been smoothed with the gravitational algorithm.
- **Sharp**: The images in this dataset are similar to those in Smoo but inverting the sense of the force over each pixel and sharpening the tonal variations.

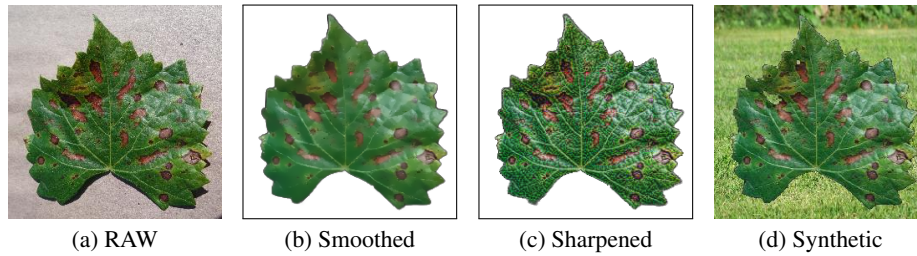


Fig. 2. Example images in PlantVillage dataset [19] from class *Grape Esca (Black Measles)* with the different preprocessings and from the PlantVillage synthetic dataset. (a) shows the original raw image, (b) is the preprocessed image applying smoothing, (c) is the preprocessed image using sharpening and (d) is the generated image adding grass background

The dataset and its derivatives were divided into three subsets with an 80/20 ratio, 80% for training, 10% for testing and the remaining 10% for validation.

Furthermore, to verify the robustness of each of the tested configurations, we trained them with a modified version of the PlantVillage dataset, where the uniform background was removed and replaced with a field image. In this way, more realistic images are simulated. This modified dataset introduced in [15] is called synthetic PlantVillage dataset (Synth-PV).

To interpret the results obtained in the confusion matrices, where True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) are extracted, we quantify the results using the following well-known Precision ($Prec$), Recall (Rec) and F_β measures:

$$Prec = \frac{TP}{TP + FP}, Rec = \frac{TP}{TP + FN}, F_\beta = (1 + \beta^2) \frac{Prec \cdot Rec}{\beta^2 \cdot Prec + Rec}.$$

We select the values of $\beta = 0.5$ and $\beta = 1$ as the most commonly used in the literature. In particular, The F_1 score balances precision and recall, making it valuable when both measures are important. Taking the harmonic mean of precision and recall provides a more comprehensive assessment of a model's performance, especially in tasks where false positives and false negatives have distinct consequences, such as medical diagnosis or plant disease detection. In addition, we use Accuracy and Loss metrics for the training and validation phases.

Combining all the six selected networks, we are to analyse along with the possible parameters that we configure (transfer learning and data augmentation) with two possibilities each one and the four different sets generated from the original PlantVillage dataset,

we have 96 different configurations. To facilitate analysis and visualisation of the results, we separate the 96 experiments into two blocks separated by whether they are mobile-orientated network architectures (Inception v3, MobileNetV2, NASNetMobile), shown in Table 1, or high-performance-computer-orientated (EfficientNet-based), presented in Table 2.

5.2. Discussion

In this subsection, we expose and analyse the results obtained with the training and testing with the different network models over the original PlantVillage dataset with the proposed preprocessing step in the first phase. We then show the test results obtained with the previously trained models in the synthetic PlantVillage dataset.

We show the different results of the train, validation, and test in Tables 1-2. We can see that the accuracy obtained during the training phase is very similar and relatively high for all the experiments, going from 0.850 to 0.993. This great value decays as validation is performed. In this case, we obtain values between 0.578 and 0.992. The results indicate that all those networks prepared for mobile devices get lower results, which is the expected behaviour, as they are optimised for limited-power devices. Despite their optimized architecture designed for limited devices, these models perform quite well. For instance, NasNetMobile achieves an accuracy validation score of 0.968, which is not too far behind the top performer, EfficientNetV2, with a validation accuracy of 0.992. As a general overview, we can extract from the training results, on the one hand, that fine-tuning (Ft) and data augmentation are not beneficial if we look at the training accuracy but permits us to retain it in the validation step for the mobile-oriented Nets. On the other hand, when viewing EfficientNet experiments, we can see that data augmentation does not benefit the training and validation accuracy, as the best results are obtained only by fine-tuning. During the training and validation phase, the best performers are E_{63} and E_{64} , that is, EfficientNet-B0 and EfficientNetV2-B0 using fine-tuning but not data augmentation.

If we observe more locally at each network type, for Inception v3, the best performance in terms of validation accuracy is E_2 , using the Raw data set and without fine-tuning or data augmentation. The same behaviour happens with MobileNetV2, but in the case of NASNetMobile, the best performer in this terms is also obtained in the Raw dataset but using fine-tuning and data augmentation. Regarding EfficientNet results, the best performer configuration using EfficientNet-B0 uses fine-tuning and data augmentation over the Raw dataset, just the opposite to mobile-orientated designs. When observing both EfficientNetV2-B0 with transfer learning from a subset of Imagenet and the complete one, we can see that we have to use fine-tuning but not data augmentation to obtain the best result.

Once the models are trained, we evaluate them during the test phase obtaining the quantitative results in Tables 3-4 with the measures indicated in Section 5.1. We tested our trained models with the corresponding test partition images of each model, and to analyse its robustness to real field images, we also tested each model with the Synthetic-PV dataset. As can be seen, the best performers with the original dataset usually decay with the synthetic one. For example, in the case of experiments with mobile-oriented nets ($E_1 - E_{48}$), the best performer in terms of original images is E_{21} , but its performance decays by approximately 76%, and the best one in terms of real synthetic images is E_{17}

maintaining its performance with a decay 24 %. This tells us it is more robust to interference from the leaf context. The same behaviour occurs in the case of EfficientNet-based experiments. The best performer is not robust and loses 76% of its performance. Instead, the best performance in the Synthetic dataset has a decay of 25 %.

In a more detailed analysis of the test results, looking at ∇F_1 by network type, we can see that with Inception v3 and MobileNetV2, excluding E_{14} due to its low results in both the original and synthetic data sets, the most stable configurations are E_2 and E_{18} , which match the results of the training phase. In the case of NASNetMobile, the stables experiment is E_{42} , which is not among the best performers in the training phase. Additionally, the best performer using NASNetMobile during training decays 46%.

On the other side, as for EfficientNet-B0 experiments, the more robust configuration is E_{75} , which has been trained over the Mixed dataset. However, close to the most stable experiment, we find those who have been trained with Smoo and Sharp preprocessing, such as E_{81} or E_{60} , respectively. In the case of EfficientNetV2-B0 with transfer learning with the Imagenet subset, we obtain slightly less stable results, being the best E_{61} , that is, using the Mixed dataset. With this network family, all other configurations have more than 50% performance variation. Finally, looking at the EfficientNetV2-B0 with the complete Imagenet transfer learning, the most stable configuration is E_{74} , trained with the Mixed dataset. Then, With a nearly a 30% decay in the F_1 measure, we have configurations trained with Smoo preprocessing (E_{56} and E_{80}).

As a result of the database validation process, a series of metrics were extracted for each of the experiments conducted to determine which of the experiments and which network architecture is the best for classifying plant diseases. Figure 3 shows a plot of the Nightingale rose, which is a plot on a polar coordinate grid. This radial graph divided each of the 48 experiments into equal segments. The distance of each segment from the centre of the polar axis depends on the value of each score it represents. In this way, each of the rings from the centre of the polar grid represents a higher value of each score. Four colours will be used to differentiate each of the sockets in the following graphs, the blue colour represents the F_1 score of the models, the yellow colour represents the $F_{0.5}$, the green colour represents the *Recall*, and the red colour represents the *Precision*.

Figure 3 shows the scores obtained from the 96 experiments with the different networks. Figure 3 (a) shows the scores for the networks launched on Edge devices: MobileNet, Inception v3 and NASNetMobile. Figure 3 (b) shows the scores obtained for EfficientNet networks, which can be executed on devices with higher-performance computers. As shown in Figure 3 (a), the scores obtained are very unstable compared to those in Figure 3 (b).

Although both networks used the same dataset, the internal architectures of each network make a difference.

To sum up, in this work we have analysed different network architectures and configurations for the automatic detection of plant disease. We have also tested our trained models, evaluating their robustness and behaviour with different image alterations and synthetic images that simulate a real environment.

The results obtained have shown in a primal analysis that the best performers during the training phase tend to have a significant decay in the test phase on real-context simulated images. Therefore, models with near-the-top best performers but slightly more restrained results are more stable when used in real scenarios. We have also noticed that

#	Network	Ft	Da	Data	Accuracy T	Accuracy V	Loss T	Loss V
E_1	InceptionV3	✗	✗	Mixed	.967	.858	.122	.500
E_2	InceptionV3	✗	✗	Raw	.940	.924	.212	.279
E_3	InceptionV3	✗	✗	Sharp	.917	.905	.277	.325
E_4	InceptionV3	✗	✗	Smoo	.939	.915	.213	.297
E_5	InceptionV3	✓	✗	Mixed	.975	.825	.285	.962
E_6	InceptionV3	✓	✗	Raw	.979	.578	.300	.909
E_7	InceptionV3	✓	✗	Sharp	.973	.844	.327	.889
E_8	InceptionV3	✓	✗	Smoo	.971	.902	.323	.569
E_9	InceptionV3	✗	✓	Mixed	.922	.811	.261	.630
E_{10}	InceptionV3	✗	✓	Raw	.867	.884	.435	.389
E_{11}	InceptionV3	✗	✓	Sharp	.850	.872	.494	.418
E_{12}	InceptionV3	✗	✓	Smoo	.876	.912	.414	.300
E_{13}	InceptionV3	✓	✓	Mixed	.963	.790	.328	.247
E_{14}	InceptionV3	✓	✓	Raw	.968	.735	.341	.368
E_{15}	InceptionV3	✓	✓	Sharp	.957	.679	.383	.951
E_{16}	InceptionV3	✓	✓	Smoo	.955	.909	.383	.549
E_{17}	MobileNetV2	✗	✗	Mixed	.988	.908	.062	.315
E_{18}	MobileNetV2	✗	✗	Raw	.980	.966	.095	.138
E_{19}	MobileNetV2	✗	✗	Sharp	.966	.941	.136	.220
E_{20}	MobileNetV2	✗	✗	Smoo	.974	.953	.109	.176
E_{21}	MobileNetV2	✓	✗	Mixed	.980	.878	.178	.533
E_{22}	MobileNetV2	✓	✗	Raw	.979	.940	.196	.360
E_{23}	MobileNetV2	✓	✗	Sharp	.974	.958	.210	.279
E_{24}	MobileNetV2	✓	✗	Smoo	.976	.953	.207	.326
E_{25}	MobileNetV2	✗	✓	Mixed	.961	.891	.144	.354
E_{26}	MobileNetV2	✗	✓	Raw	.938	.939	.220	.221
E_{27}	MobileNetV2	✗	✓	Sharp	.915	.926	.291	.271
E_{28}	MobileNetV2	✗	✓	Smoo	.930	.942	.242	.206
E_{29}	MobileNetV2	✓	✓	Mixed	.969	.853	.217	.655
E_{30}	MobileNetV2	✓	✓	Raw	.972	.959	.229	.257
E_{31}	MobileNetV2	✓	✓	Sharp	.964	.797	.247	.057
E_{32}	MobileNetV2	✓	✓	Smoo	.965	.933	.252	.377
E_{33}	NasNetMobile	✗	✗	Mixed	.972	.871	.122	.451
E_{34}	NasNetMobile	✗	✗	Raw	.948	.930	.203	.260
E_{35}	NasNetMobile	✗	✗	Sharp	.928	.915	.264	.309
E_{36}	NasNetMobile	✗	✗	Smoo	.949	.939	.206	.241
E_{37}	NasNetMobile	✓	✗	Mixed	.984	.903	.323	.625
E_{38}	NasNetMobile	✓	✗	Raw	.985	.943	.336	.503
E_{39}	NasNetMobile	✓	✗	Sharp	.980	.957	.360	.430
E_{40}	NasNetMobile	✓	✗	Smoo	.983	.952	.350	.477
E_{41}	NasNetMobile	✗	✓	Mixed	.931	.815	.240	.621
E_{42}	NasNetMobile	✗	✓	Raw	.898	.880	.363	.421
E_{43}	NasNetMobile	✗	✓	Sharp	.874	.893	.429	.385
E_{44}	NasNetMobile	✗	✓	Smoo	.900	.925	.357	.284
E_{45}	NasNetMobile	✓	✓	Mixed	.978	.870	.351	.729
E_{46}	NasNetMobile	✓	✓	Raw	.979	.968	.364	.396
E_{47}	NasNetMobile	✓	✓	Sharp	.971	.935	.389	.503
E_{48}	NasNetMobile	✓	✓	Smoo	.971	.954	.391	.446

Table 1. PlantVillage experiments with 7 Epochs, feature vector model using InceptionV3, MobileNetV2 and NasNetMobile networks. Fine-tuning (Ft) and Data augmentation (Da) use are indicated with a cross (✗) and a tick (✓), differentiating the train (T) and validation (V) values. The maximum values for Accuracy and minimum values for Loss are shown in bold

#	Network	Ft	Da	Data	Accuracy T	Accuracy V	Loss T	Loss V
<i>E</i> ₄₉	EfficientnetV2-B0-imagenet1k	✗	✗	Mixed	.989	.926	.065	.254
<i>E</i> ₅₀	EfficientnetV2-B0-imagenet21k-ft1k	✗	✗	Mixed	.991	.947	.045	.186
<i>E</i> ₅₁	Efficientnet-B0	✗	✗	Mixed	.993	.946	.052	.193
<i>E</i> ₅₂	EfficientnetV2-B0-imagenet1k	✗	✗	Raw	.984	.976	.087	.114
<i>E</i> ₅₃	EfficientnetV2-B0-imagenet21k-ft1k	✗	✗	Raw	.987	.980	.064	.091
<i>E</i> ₅₄	Efficientnet-B0	✗	✗	Raw	.990	.983	.071	.091
<i>E</i> ₅₅	EfficientnetV2-B0-imagenet1k	✗	✗	Smoo	.975	.967	.124	.146
<i>E</i> ₅₆	EfficientnetV2-B0-imagenet21k-ft1k	✗	✗	Smoo	.983	.976	.079	.103
<i>E</i> ₅₇	Efficientnet-B0	✗	✗	Smoo	.980	.978	.102	.119
<i>E</i> ₅₈	EfficientnetV2-B0-imagenet1k	✗	✗	Sharp	.973	.958	.127	.174
<i>E</i> ₅₉	EfficientnetV2-B0-imagenet21k-ft1k	✗	✗	Sharp	.978	.971	.097	.127
<i>E</i> ₆₀	Efficientnet-B0	✗	✗	Sharp	.977	.966	.115	.138
<i>E</i> ₆₁	EfficientnetV2-B0-imagenet1k	✓	✗	Mixed	.990	.946	.048	.197
<i>E</i> ₆₂	EfficientnetV2-B0-imagenet21k-ft1k	✓	✗	Mixed	.989	.958	.053	.154
<i>E</i> ₆₃	Efficientnet-B0	✓	✗	Mixed	.993	.960	.156	.286
<i>E</i> ₆₄	EfficientnetV2-B0-imagenet1k	✓	✗	Raw	.990	.992	.047	.042
<i>E</i> ₆₅	EfficientnetV2-B0-imagenet21k-ft1k	✓	✗	Raw	.989	.991	.052	.043
<i>E</i> ₆₆	Efficientnet-B0	✓	✗	Raw	.989	.986	.173	.180
<i>E</i> ₆₇	EfficientnetV2-B0-imagenet1k	✓	✗	Smoo	.988	.982	.058	.078
<i>E</i> ₆₈	EfficientnetV2-B0-imagenet21k-ft1k	✓	✗	Smoo	.987	.981	.057	.081
<i>E</i> ₆₉	Efficientnet-B0	✓	✗	Smoo	.986	.976	.183	.221
<i>E</i> ₇₀	EfficientnetV2-B0-imagenet1k	✓	✗	Sharp	.990	.981	.049	.090
<i>E</i> ₇₁	EfficientnetV2-B0-imagenet21k-ft1k	✓	✗	Sharp	.987	.969	.058	.141
<i>E</i> ₇₂	Efficientnet-B0	✓	✗	Sharp	.988	.975	.182	.216
<i>E</i> ₇₃	EfficientnetV2-B0-imagenet1k	✗	✓	Mixed	.970	.920	.122	.264
<i>E</i> ₇₄	EfficientnetV2-B0-imagenet21k-ft1k	✗	✓	Mixed	.978	.923	.090	.267
<i>E</i> ₇₅	Efficientnet-B0	✗	✓	Mixed	.975	.915	.106	.298
<i>E</i> ₇₆	EfficientnetV2-B0-imagenet1k	✗	✓	Raw	.962	.961	.160	.166
<i>E</i> ₇₇	EfficientnetV2-B0-imagenet21k-ft1k	✗	✓	Raw	.969	.964	.123	.138
<i>E</i> ₇₈	Efficientnet-B0	✗	✓	Raw	.969	.963	.134	.148
<i>E</i> ₇₉	EfficientnetV2-B0-imagenet1k	✗	✓	Smoo	.943	.949	.218	.198
<i>E</i> ₈₀	EfficientnetV2-B0-imagenet21k-ft1k	✗	✓	Smoo	.958	.965	.155	.136
<i>E</i> ₈₁	Efficientnet-B0	✗	✓	Smoo	.957	.965	.176	.150
<i>E</i> ₈₂	EfficientnetV2-B0-imagenet1k	✗	✓	Sharp	.942	.939	.222	.229
<i>E</i> ₈₃	EfficientnetV2-B0-imagenet21k-ft1k	✗	✓	Sharp	.950	.942	.176	.202
<i>E</i> ₈₄	Efficientnet-B0	✗	✓	Sharp	.947	.954	.208	.174
<i>E</i> ₈₅	EfficientnetV2-B0-imagenet1k	✓	✓	Mixed	.984	.938	.069	.220
<i>E</i> ₈₆	EfficientnetV2-B0-imagenet21k-ft1k	✓	✓	Mixed	.982	.921	.078	.286
<i>E</i> ₈₇	Efficientnet-B0	✓	✓	Mixed	.981	.959	.198	.286
<i>E</i> ₈₈	EfficientnetV2-B0-imagenet1k	✓	✓	Raw	.986	.981	.061	.079
<i>E</i> ₈₉	EfficientnetV2-B0-imagenet21k-ft1k	✓	✓	Raw	.984	.979	.068	.096
<i>E</i> ₉₀	Efficientnet-B0	✓	✓	Raw	.984	.988	.192	.180
<i>E</i> ₉₁	EfficientnetV2-B0-imagenet1k	✓	✓	Smoo	.980	.980	.081	.081
<i>E</i> ₉₂	EfficientnetV2-B0-imagenet21k-ft1k	✓	✓	Smoo	.979	.981	.085	.087
<i>E</i> ₉₃	Efficientnet-B0	✓	✓	Smoo	.979	.978	.213	.216
<i>E</i> ₉₄	EfficientnetV2-B0-imagenet1k	✓	✓	Sharp	.978	.952	.086	.166
<i>E</i> ₉₅	EfficientnetV2-B0-imagenet21k-ft1k	✓	✓	Sharp	.977	.940	.092	.206
<i>E</i> ₉₆	Efficientnet-B0	✓	✓	Sharp	.978	.957	.215	.297

Table 2. PlantVillage experiments with 7 Epochs, feature vector model using EfficientNet-B0, EfficientNetV2-B0 pre-trained with imagenet-ilsvrc-2012-cls and EfficientNetV2-B0 pre-trained with full ImageNet and fine-tuned with imagenet1k networks. Fine-tuning (Ft) and Data augmentation (Da) use are indicated with a cross (✗) and a tick (✓), differentiating the train (T) and validation (V) values. The maximum values for Accuracy and minimum values for Loss are shown in bold

#	<i>Prec.</i>	<i>Prec.</i> '	∇ <i>Prec.</i>	<i>Rec.</i>	<i>Rec.</i> '	∇ <i>Rec.</i>	$F_{0.5}$	$F'_{0.5}$	$\nabla F_{0.5}$	F_1	F'_1	∇F_1
E_1	.901	.643	.258	.917	.538	.379	.902	.552	.350	.905	.519	.386
E_2	.807	.733	.074	.843	.691	.152	.806	.699	.107	.812	.681	.131
E_3	.893	.652	.241	.884	.479	.405	.888	.516	.372	.884	.461	.423
E_4	.907	.612	.295	.879	.496	.383	.893	.520	.373	.883	.476	.407
E_5	.849	.579	.270	.681	.341	.340	.724	.356	.368	.673	.304	.369
E_6	.855	.502	.353	.860	.296	.564	.843	.285	.558	.838	.245	.593
E_7	.911	.493	.418	.857	.268	.589	.887	.281	.606	.866	.237	.629
E_8	.862	.527	.334	.817	.291	.526	.833	.294	.538	.812	.247	.565
E_9	.878	.645	.232	.881	.495	.386	.876	.525	.351	.875	.475	.400
E_{10}	.813	.694	.119	.827	.568	.259	.805	.586	.219	.802	.549	.253
E_{11}	.878	.654	.223	.844	.495	.349	.866	.541	.324	.854	.484	.370
E_{12}	.890	.647	.243	.876	.505	.371	.884	.529	.355	.879	.486	.393
E_{13}	.915	.314	.601	.894	.209	.685	.901	.175	.726	.890	.164	.726
E_{14}	.709	.637	.071	.504	.399	.104	.547	.426	.121	.479	.371	.107
E_{15}	.883	.549	.333	.862	.215	.647	.866	.264	.602	.856	.192	.663
E_{16}	.915	.318	.597	.915	.181	.734	.902	.122	.780	.899	.109	.790
E_{17}	.958	.709	.249	.968	.574	.394	.960	.592	.368	.963	.552	.410
E_{18}	.881	.744	.137	.904	.665	.239	.882	.669	.212	.886	.644	.242
E_{19}	.933	.692	.241	.939	.475	.463	.932	.489	.443	.932	.442	.490
E_{20}	.957	.612	.345	.947	.496	.450	.954	.490	.463	.951	.457	.493
E_{21}	.980	.429	.550	.982	.255	.727	.980	.264	.716	.980	.221	.759
E_{22}	.829	.616	.212	.852	.465	.386	.813	.456	.356	.807	.423	.384
E_{23}	.962	.428	.534	.934	.305	.629	.952	.257	.695	.942	.235	.707
E_{24}	.851	.338	.512	.802	.147	.655	.814	.143	.670	.792	.117	.675
E_{25}	.940	.696	.244	.939	.503	.435	.939	.567	.372	.938	.507	.430
E_{26}	.881	.719	.162	.897	.650	.247	.881	.643	.238	.883	.623	.260
E_{27}	.904	.641	.263	.899	.406	.493	.900	.486	.414	.897	.415	.482
E_{28}	.940	.611	.328	.935	.510	.425	.937	.494	.443	.935	.468	.467
E_{29}	.974	.481	.493	.974	.323	.651	.973	.266	.707	.973	.249	.724
E_{30}	.852	.658	.193	.862	.375	.487	.836	.419	.417	.831	.358	.473
E_{31}	.868	.422	.446	.846	.222	.624	.839	.214	.625	.823	.187	.635
E_{32}	.952	.196	.756	.926	.124	.802	.941	.105	.836	.932	.087	.845
E_{33}	.915	.570	.345	.922	.451	.471	.915	.446	.469	.917	.410	.507
E_{34}	.813	.630	.182	.851	.553	.297	.812	.558	.254	.816	.531	.284
E_{35}	.912	.540	.372	.909	.395	.514	.909	.363	.546	.907	.341	.566
E_{36}	.931	.578	.353	.905	.398	.507	.922	.369	.553	.913	.339	.574
E_{37}	.946	.671	.274	.927	.404	.523	.936	.440	.496	.927	.377	.550
E_{38}	.951	.632	.318	.943	.343	.599	.944	.368	.576	.940	.307	.633
E_{39}	.951	.430	.520	.922	.243	.679	.941	.241	.700	.930	.210	.720
E_{40}	.944	.138	.805	.928	.098	.830	.936	.059	.877	.929	.051	.878
E_{41}	.882	.542	.339	.891	.391	.500	.882	.374	.508	.884	.336	.548
E_{42}	.787	.632	.155	.810	.535	.275	.780	.527	.253	.779	.504	.275
E_{43}	.892	.538	.354	.880	.416	.464	.886	.397	.489	.881	.376	.505
E_{44}	.914	.590	.324	.891	.451	.440	.906	.428	.478	.896	.403	.493
E_{45}	.968	.417	.550	.978	.290	.688	.969	.289	.679	.972	.255	.717
E_{46}	.916	.672	.244	.927	.490	.437	.913	.492	.421	.914	.449	.465
E_{47}	.886	.445	.441	.853	.213	.640	.861	.228	.633	.841	.188	.653
E_{48}	.965	.541	.423	.938	.244	.694	.958	.256	.702	.949	.221	.728

Table 3. Resulting test performance of the configurations trained with the parameters in Table 1 over the original data set and its pre-processed variants along with those obtained with the Synth-PV dataset and its difference. The maximum values for *Prec.*, *Rec* and F_β values and minimum values for ∇ are shown in bold

#	<i>Prec.</i>	<i>Prec.</i> '	∇ <i>Prec.</i>	<i>Rec.</i>	<i>Rec.</i> '	∇ <i>Rec.</i>	$F_{0.5}$	$F'_{0.5}$	$\nabla F_{0.5}$	F_1	F'_1	∇F_1
E_{49}	.893	.591	.302	.835	.408	.427	.872	.412	.460	.851	.369	.482
E_{50}	.912	.803	.108	.887	.666	.220	.904	.699	.205	.895	.659	.236
E_{51}	.931	.733	.198	.909	.619	.290	.924	.608	.316	.917	.578	.339
E_{52}	.973	.667	.305	.969	.408	.560	.972	.424	.548	.971	.373	.598
E_{53}	.971	.814	.157	.974	.709	.265	.971	.707	.264	.972	.685	.286
E_{54}	.976	.700	.276	.968	.600	.368	.973	.607	.366	.970	.575	.395
E_{55}	.956	.651	.304	.953	.472	.481	.955	.443	.512	.954	.416	.538
E_{56}	.971	.777	.193	.968	.677	.290	.970	.695	.275	.969	.659	.309
E_{57}	.970	.708	.262	.965	.574	.391	.968	.559	.408	.966	.532	.433
E_{58}	.949	.639	.309	.933	.416	.517	.943	.388	.554	.937	.365	.572
E_{59}	.965	.776	.188	.960	.541	.418	.963	.614	.349	.961	.542	.418
E_{60}	.955	.678	.276	.944	.558	.385	.951	.535	.415	.947	.512	.434
E_{61}	.926	.779	.147	.886	.573	.313	.911	.646	.265	.897	.583	.314
E_{62}	.931	.558	.373	.909	.288	.621	.918	.341	.577	.910	.282	.628
E_{63}	.940	.734	.205	.931	.566	.365	.935	.608	.327	.931	.549	.382
E_{64}	.990	.700	.290	.990	.426	.564	.990	.462	.528	.990	.404	.586
E_{65}	.991	.472	.519	.989	.266	.723	.991	.269	.722	.990	.222	.768
E_{66}	.986	.586	.400	.984	.313	.671	.985	.318	.667	.984	.285	.699
E_{67}	.982	.382	.600	.982	.146	.836	.981	.177	.804	.981	.141	.840
E_{68}	.985	.556	.428	.979	.258	.721	.983	.273	.710	.981	.220	.761
E_{69}	.969	.551	.417	.960	.328	.631	.966	.381	.585	.963	.320	.643
E_{70}	.976	.384	.592	.967	.238	.729	.973	.260	.713	.969	.223	.746
E_{71}	.967	.494	.473	.969	.142	.827	.966	.151	.815	.966	.105	.861
E_{72}	.965	.707	.258	.969	.434	.534	.965	.481	.484	.966	.420	.546
E_{73}	.889	.592	.297	.853	.443	.410	.877	.423	.454	.864	.392	.472
E_{74}	.910	.788	.122	.849	.654	.194	.885	.679	.205	.863	.643	.219
E_{75}	.879	.712	.167	.826	.566	.260	.859	.554	.304	.839	.524	.314
E_{76}	.952	.642	.309	.945	.473	.472	.949	.461	.487	.946	.422	.524
E_{77}	.960	.804	.155	.962	.717	.245	.959	.734	.224	.959	.704	.255
E_{78}	.961	.758	.202	.951	.608	.342	.958	.626	.331	.954	.588	.366
E_{79}	.949	.575	.374	.914	.432	.482	.937	.381	.556	.925	.359	.566
E_{80}	.958	.756	.201	.960	.646	.313	.958	.652	.305	.957	.617	.339
E_{81}	.954	.687	.266	.949	.539	.409	.952	.569	.383	.949	.526	.422
E_{82}	.933	.527	.406	.896	.374	.522	.917	.360	.557	.905	.339	.566
E_{83}	.939	.722	.216	.927	.495	.432	.931	.568	.363	.925	.501	.424
E_{84}	.951	.638	.312	.941	.462	.478	.948	.482	.465	.944	.449	.494
E_{85}	.909	.618	.291	.882	.391	.491	.896	.423	.473	.885	.374	.511
E_{86}	.909	.598	.311	.845	.355	.490	.884	.379	.505	.860	.332	.528
E_{87}	.940	.798	.141	.926	.433	.493	.935	.536	.399	.929	.449	.480
E_{88}	.972	.758	.213	.964	.461	.502	.965	.485	.480	.962	.425	.536
E_{89}	.976	.564	.412	.975	.408	.567	.975	.410	.565	.974	.362	.612
E_{90}	.984	.661	.322	.990	.401	.589	.984	.455	.528	.986	.400	.586
E_{91}	.979	.245	.734	.973	.144	.829	.977	.120	.857	.975	.096	.879
E_{92}	.982	.349	.633	.979	.168	.810	.981	.156	.825	.980	.141	.839
E_{93}	.972	.467	.504	.969	.241	.728	.970	.259	.711	.969	.218	.751
E_{94}	.929	.650	.279	.941	.279	.661	.927	.330	.597	.928	.281	.647
E_{95}	.930	.606	.324	.930	.310	.620	.921	.348	.573	.918	.290	.628
E_{96}	.930	.574	.356	.921	.259	.662	.922	.298	.624	.917	.245	.672

Table 4. Resulting test performance of the configurations trained with the parameters in Table 2 over the original data set and its pre-processed variants along with those obtained with the Synth-PV dataset and its difference. The maximum values for *Prec.*, *Rec* and F_β values and minimum values for ∇ are shown in bold

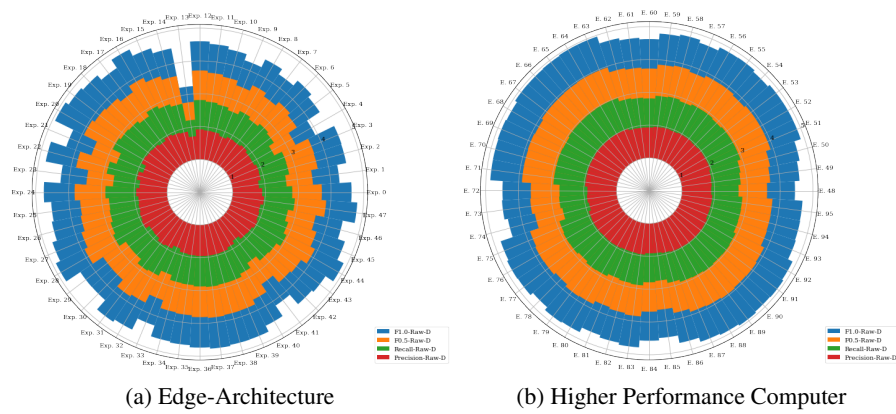


Fig. 3. Performance results obtained by each of the 96 experiment configurations with the PlantVillage datasets and its pre-processed variants

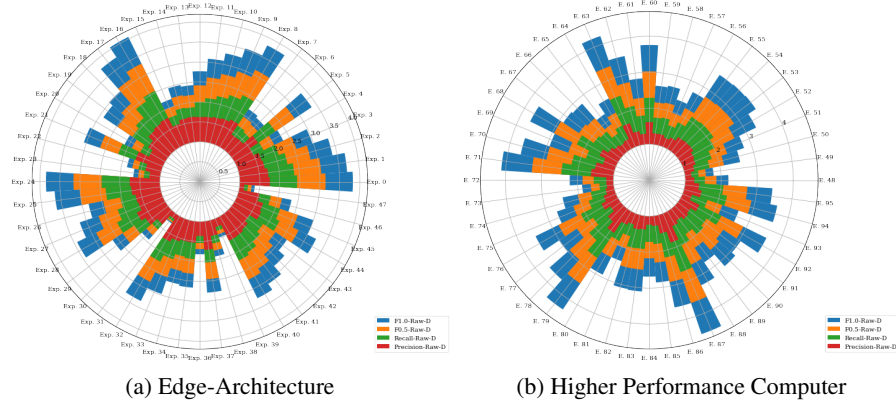


Fig. 4. Performance results obtained by each of the 96 experiment configurations with the synthetic PlantVillage dataset

those configurations trained over preprocessed dataset tend to be more stable and robust during the validation process with the simulated data.

6. Conclusions

In this work, we have analyzed different network architectures and configurations for the automatic detection of plant disease. We have shed light on several important aspects while acknowledging its inherent limitations. We have recognized potential biases in the dataset used, stemming from factors like geographical and temporal distribution of diseases, which can influence model performance. Furthermore, the complex real-world context of plant disease management introduces confounding factors such as weather conditions, soil quality, and agricultural practices, which must be considered when interpreting our results.

Throughout our research, we have grappled with the trade-offs between precision and recall, recognizing that achieving high precision may entail the risk of missing some cases while emphasizing recall can lead to more false alarms. These trade-offs underscore the need for a nuanced approach that can be tailored to specific agricultural scenarios and disease management strategies.

While subject to these limitations, our findings hold significant promise for practical applications in agriculture and plant disease management. The deep learning models and strategies identified can be valuable tools in real-world scenarios. They can facilitate early disease detection, providing farmers with timely information for proactive intervention. Moreover, the analyzed models can contribute to precision farming by pinpointing affected areas within fields, thus reducing the indiscriminate use of pesticides and minimizing environmental impact. This aligns with the global push toward sustainable agriculture, where minimizing chemical inputs is a key goal.

Looking ahead, our research opens doors to further exploration. Addressing dataset biases through more extensive and diverse data collection efforts can result in more robust models with broader applicability. Investigating the generalization of our models to different geographical regions and agricultural settings is another promising avenue. Additionally, interdisciplinary collaboration with plant pathology, agriculture, and environmental science experts can enhance the practicality and real-world relevance of our disease detection models. Developing interactive systems that involve farmers and agricultural experts in the decision-making process is another avenue to explore, allowing for more context-aware disease management.

Acknowledgments. The authors gratefully acknowledge the financial support of Conselleria d'Innovació, Universitat, Ciència i Societat Digital from Comunitat Valenciana (APOSTD/2021/227), the European Social Fund (Investing In Your Future), grant PID2021-123673OB-C31 funded by MCIN/AEI/ 10.13039/501100011033 and by "ERDF A way of making Europe" and grant from the Research Services of Universitat Politècnica de València (PAID-PD-22).

References

1. Atila, Ü., Uçar, M., Akyol, K., Uçar, E.: Plant leaf disease classification using efficientnet deep learning model. *Ecological Informatics* 61, 101182 (2021)
2. Atila, U., Uçar, M., Akyol, K., Uçar, E.: Plant leaf disease classification using EfficientNet deep learning model. *Ecological Informatics* 61, 101182 (2021)
3. Barbedo, J.G.A.: A review on the main challenges in automatic plant disease identification based on visible range images. *Biosystems engineering* 144, 52–60 (2016)
4. Bezdek, J.C., Chandrasekhar, R., Attikouzel, Y.: A geometric approach to edge detection. *IEEE Transactions on Fuzzy Systems* 6(1), 52–75 (1998)
5. Bhakta, I., Phadikar, S., Majumder, K., Mukherjee, H., Sau, A.: A novel plant disease prediction model based on thermal images using modified deep convolutional neural network. *Precision Agriculture* pp. 1–17 (2022)
6. Camargo, A., Smith, J.: Image pattern classification for the identification of disease causing agents in plants. *Computers and Electronics in Agriculture* 66(2), 121–125 (2009)
7. Costa, L., Nunes, L., Ampatzidis, Y.: A new visible band index (vndvi) for estimating ndvi values on rgb images utilizing genetic algorithms. *Computers and Electronics in Agriculture* 172, 105334 (2020)

8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. Devi, R., Kumar, V., Sivakumar, P.: Efficientnetv2 model for plant disease classification and pest recognition. *Computer Systems Science & Engineering* 45(2) (2023)
10. Dong, Z.: Image-Based Plant Leaf Disease Recognition with InceptionV3 Network. Ph.D. thesis, The Ohio State University (2021)
11. Elfatimi, E., Eryigit, R., Elfatimi, L.: Beans leaf diseases classification using mobilenet models. *IEEE Access* 10, 9471–9482 (2022)
12. Glegoła, W., Karpus, A., Przybyłek, A.: Mobilenet family tailored for raspberry pi. *Procedia Computer Science* 192, 2249–2258 (2021)
13. Gueye, Y., Mbaye, M.: Kmeans kernel-learning based ai-iot framework for plant leaf disease detection. In: *International Conference on Service-Oriented Computing*. pp. 549–563. Springer (2020)
14. Gui, P., Dang, W., Zhu, F., Zhao, Q.: Towards automatic field plant disease recognition. *Computers and Electronics in Agriculture* 191, 106523 (2021)
15. Gui, P., Dang, W., Zhu, F., Zhao, Q.: Towards automatic field plant disease recognition. *Computers and Electronics in Agriculture* 191, 106523 (2021)
16. Hasan, R.I., Yusuf, S.M., Alzubaidi, L.: Review of the state of the art of deep learning for plant diseases: A broad analysis and discussion. *Plants* 9(10), 1302 (2020)
17. Hassan, S.M., Maji, A.K.: Plant disease identification using a novel convolutional neural network. *IEEE Access* 10, 5390–5401 (2022)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 770–778 (2016)
19. Hughes, D., Salathé, M., et al.: An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint arXiv:1511.08060* (2015)
20. Khan, S., Narvekar, M.: Novel fusion of color balancing and superpixel based approach for detection of tomato plant diseases in natural complex environment. *Journal of King Saud University - Computer and Information Sciences* (2020)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012)
22. Lee, S.H., Goëau, H., Bonnet, P., Joly, A.: Attention-based recurrent neural network for plant disease classification. *Frontiers in Plant Science* 11, 601250 (2020)
23. Madrid, N., Lopez-Molina, C., Hurtik, P.: Non-linear scale-space based on fuzzy contrast enhancement: Theoretical results. *Fuzzy Sets and Systems* 421, 133–157 (2021)
24. Mahesh, T., Sivakami, R., Manimozhi, I., Krishnamoorthy, N., Swapna, B., et al.: Early predictive model for detection of plant leaf diseases using mobilenetv2 architecture. *International Journal of Intelligent Systems and Applications in Engineering* 11(2), 46–54 (2023)
25. Marco-Detchart, C., Lopez-Molina, C., Fernandez, J., Bustince, H.: A gravitational approach to image smoothing. In: *Advances in Fuzzy Logic and Technology 2017*, pp. 468–479. Springer (2017)
26. Mohd Noor, F.N., Mohd Isa, W.H., Khairuddin, I.M., Mohd Razman, M.A., Musa, R.M., PP Abdul Majeed, A., et al.: The diagnosis of diabetic retinopathy: An evaluation of different classifiers with the inception v3 model as a feature extractor. In: *International Conference on Robot Intelligence Technology and Applications*. pp. 392–397. Springer (2022)
27. Ngo, H., Fang, H., Wang, H.: Beamforming and scalable image processing in vehicle-to-vehicle networks. *Journal of Signal Processing Systems* pp. 1–10 (2022)
28. Nguyen, A., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 427–436 (2015)
29. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence* 12(7), 629–639 (1990)

30. Poma, X.S., Riba, E., Sappa, A.: Dense extreme inception network: Towards a robust cnn model for edge detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1923–1932 (2020)
31. Rumpf, T., Mahlein, A.K., Steiner, U., Oerke, E.C., Dehne, H.W., Plümer, L.: Early detection and classification of plant diseases with support vector machines based on hyperspectral reflectance. Computers and electronics in agriculture 74(1), 91–99 (2010)
32. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. Advances in neural information processing systems 30 (2017)
33. Samin, O.B., Omar, M., Mansoor, M.: CapPlant: a capsule network based framework for plant disease classification. PeerJ Computer Science 7, e752 (2021)
34. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)
35. Schuler, J., Romani, S., Abdel-nasser, M., Rashwan, H., Puig, D.: Reliable Deep Learning Plant Leaf Disease Classification Based on Light-Chroma Separated Branches, pp. 375–381. IOS Press (10 2021)
36. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
37. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
38. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
39. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: International Conference on Machine Learning. pp. 10096–10106. PMLR (2021)

Cedric Marco-Detchart received his PhD on Artificial Intelligence from Universidad Pública de Navarra, and is a researcher at the Valencian Research Institute for Artificial Intelligence (VRAIN), focusing on enhancing cognitive assistance systems through comparison measures, with a particular interest in computer vision, advances in psychology and neurosciences.

Jaime Andrés Rincón Arango has a PhD in Computer Science from the Universitat Politècnica de València; he is currently an assistant professor at the University of Burgos, where he teaches. Some of the research lines in which he works are multi-agent systems, IoT, IoMT, Robotics, Edge AI, Machine Learning, Virtual Reality and Mixed Reality.

Carlos Carrascosa Casamayor holds a PhD in Computer Science from the Universitat Politècnica de València, where he has been teaching and researching since 1996, currently as an Associate Professor. Some of the research lines he is working in are: multi-agent systems, virtual organisations, agreement technologies, adaptive systems, federated learning, consensus algorithms, intelligent virtual environments, affective computing, real-time systems and the so-called "serious games".

Vicente Julian is Full Professor of the Department of Computer Systems and Computing at the Polytechnic University of Valencia (UPV) since 2017. He is also Deputy Director of Research of the Department of Computer Systems and Computation and Coordinator of the Doctorate Program in Computer Science of the UPV.

Received: December 22, 2022; Accepted: October 10, 2023.

A Flexible Approach for Demand-Responsive Public Transport in Rural Areas

Pasqual Martí¹, Jaume Jordán¹, and Vicente Julian^{1,2}

¹ Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain
pasmargi@vrain.upv.es, {jjordan,vinglada}@dsic.upv.es

² Valencian Graduate School and Research Network of Artificial Intelligence, Universitat Politècnica de València, Camí de Vera s/n, 46022 Valencia, Spain

Abstract. Rural mobility research has been left aside in favor of urban transportation. Rural areas' low demand, the distance among settlements, and an older population on average make conventional public transportation inefficient and costly. This paper assesses the contribution that on-demand mobility has the potential to make to rural areas. First, demand-responsive transportation is described, and the related literature is reviewed to gather existing system configurations. Next, we describe and implement a proposal and test it on a simulation basis. The results show a clear potential of the demand-responsive mobility paradigm to serve rural demand at an acceptable quality of service. Finally, the results are discussed, and the issues of adoption rate and input data scarcity are addressed.

Keywords: Demand-responsive, Transportation, Rural mobility, Simulation.

1. Introduction

Demand-responsive transportation (DRT) was first developed in the UK in the 1960s as a means of rural transportation [22] with a flexible route and dial-a-ride program. In the past, it has been utilized to provide on-demand transportation services for those who are physically disabled. These early initiatives depended on government money, and if that funding was cut off, they eventually ceased to exist. In fact, funding has always been a major problem in DRT because, typically, a transportation mode's flexibility results in greater operational costs [8,10]. Public transportation companies have rekindled their interest in DRT systems in today's environment of dial-a-ride private transportation [11] (taxi, Uber, Cabify) powered by smartphones and applications. The reason is twofold: On the one hand, the technological advancements in computation and electronics make it possible to solve complex problems such as online vehicle scheduling, routing and detouring in brief computational times. Moreover, the popularization of smartphones has made on-demand mobility more accessible than ever for the newer generations. Finally, the advances in autonomous mobility made demand-responsive transportation more promising. On the other hand, the flexibility and responsiveness of DRT are intuitively good attributes for an environmentally conscious, more sustainable transportation mode that may be able to reduce empty-vehicle displacement, thus reducing energy consumption and greenhouse gas emissions.

The interest of the research community in DRT has been rising in the last few years, although most of the studied and proposed systems are developed for high-density urban

areas. In contrast, the application of DRT solutions to rural settlements or areas is less explored. Rural areas count with scattered residents, a low level of transportation demand, and, on average, an older population with respect to urban areas. Its usual transportation methods feature a single line with a mid-to-high capacity vehicle and a low frequency. The lack of quality public transportation is reflected in the usage of individual motorized transport, which is the most popular form of transportation in some rural areas [24]. DRT seems appropriate to fit rural demand and has the potential to cut operating costs while being more sustainable thanks to its on-demand activation. In addition, passenger experience could be improved by lower waiting and riding times.

There are a few works that analyze the potential of DRT for rural mobility. The authors of [6,21] propose the replacement of the traditional transportation services of specific rural areas with a DRT alternative. Both works find a better overall efficiency with DRT compared to the fixed service. Particularly, the results in [6] show a decrease in the amount of traveled kilometers, operational costs, and greenhouse gas emissions per passenger. Other analytical works such as [30,1] focus on the adoption rate of these services among rural inhabitants. Their findings show a potential niche market for DRT transportation and explicit relevant factors that the user takes into account to switch to a new transportation service. Finally, the work in [23] goes over rural DRT services from a customer satisfaction perspective, evidencing a concerning conflict between user expectations and the actual system operation. The authors underscore the importance of the analysis of the rural area and the characterization of its potential customer needs for a successful DRT application. All the research cited above shows that several authors from different contexts find the use of DRT as a potential solution for improved rural mobility. However, there is a noticeable lack of papers that bring more intelligent techniques to rural mobility.

Urban areas have always had a steady flow of quality proposals, such as [16,31], focused on optimizing their processes. However, rural areas find a clear lack of proposals. Specifically, our current research is motivated by the literature gap regarding the application of intelligent techniques for rural DRT services. The main objective of this line of work is the development of practical solutions for dynamic, flexible, reliable, and economically viable rural mobility. Working on such a goal, this paper characterizes DRT systems, assessing each of the challenges their design and implementation implies. Given the specific issues of rural areas, we theorize that the DRT paradigm might be a good fit to provide displacement services to their inhabitants. We prove our hypothesis by describing and implementing a system, which is later tested by simulating its operation in a real rural area. The results show the system achieves a good quality of service over a wide area with a reduced fleet of smaller (with respect to public buses) vehicles. Our work contributes to the rural mobility research field with the introduction of an algorithm that schedules both the static and online operation of the proposed DRT service. In addition, our results show the potential DRT has to modernize and improve rural transportation systems.

This work is an extended version of the paper “Demand-responsive Mobility for Rural Areas: A Review” [19], presented at the 20th International Conference on Practical Applications of Agents and Multiagent Systems (PAAMS 2022). The rest of the paper is structured as follows. Section 2 dissects DRT through the review of relevant literature works. Then, Section 3 describes the proposed system, its components, and the algorithms that make it work. Section 4 presents the use case and the simulation results. Section 5

discusses the introduction of DRT to rural areas in accordance with our results. Finally, Section 6 concludes the work and states possible future directions for our research.

2. Demand-responsive Transportation Description

A DRT system is composed of a series of subsystems, each in charge of solving one of the many challenges a transportation system involves. These subsystems are highly configurable and can be adapted to the concrete mobility needs of a specific area. Because of that, the variety of DRT services is vast. Nevertheless, all of them deal with a concrete set of issues presented below:

- *Planning of services and scheduling of requests.* Whether it is performed in advance or in real-time after receiving transportation requests, a DRT operator must plan the operation of its fleet according to its resources. Depending on the type of system, such planning may include routing and stop assignment. In addition, in a request-based system, passengers must be assigned to a vehicle (or a concrete line) that will serve them. This assignment implies the rescheduling of the vehicle planning to include new customers while worsening as little as possible other passengers' experiences.
- *Optimizing fleet resources.* The goal is to select the appropriate vehicles with a concrete capacity such that the operation of the DRT system yields an acceptable quality of service while being economically viable and sustainable.
- *Demand prediction and estimation* can be a complementary feature of DRT systems used to optimize their operation. Such a feature can be implemented based on historical data or prediction techniques to control future and current demand. Many solutions require the passengers to explicitly state their desire to use the service by issuing a request.
- *Validation* through the definition of appropriated metrics to evaluate and compare different configurations.

Solutions to the above issues are dependent on the concrete type of DRT system that will be implemented in addition to the modeling and optimization techniques used for that. Following, we describe the different characteristics that a DRT system can have (Section 2.1) and the techniques that have been observed in the literature for their implementation (Section 2.2). Finally, we enumerate the optimization perspectives of the reviewed material (Section 2.3).

2.1. System Types

DRT systems have a series of standard elements present in all of them. Different authors apply different labels to those elements. For the current section, we have followed the terminology described in this survey [28].

In a DRT system, a *service* is the departure of a vehicle to serve the transportation requests it has assigned. One service is generally tied to a concrete area or line the transport will follow. In contrast, a *route* is the concrete path the vehicle follows, connecting all the pickups and drop-offs. A route does not necessarily include all existing stops in a line

or area. Customers are picked up and dropped off in a predefined set of *stops* within the serviced area or line. Alternatively, a *door-to-door* service can be offered, in which any user-specified location within a particular area may act as a stop. This type of mobility is thought to be *shared*; i.e.: multiple customers are served by the same vehicle. Typical vehicle choices for demand-responsive services include a taxi-like car with a capacity of 4 passengers, vans with 8 to 12 seats, and mini-buses or buses with 16 to 22 seats, respectively.

Many use cases exist for demand-responsive transportation. Specifically, for rural DRT, we find the following: transportation within rural settlements, transportation between rural settlements, and transportation between rural and urban settlements. In practice, these cases can be reduced to two systems: *many-to-many*, with multiple origins and destination locations, and *many-to-one*, where origin and destination locations share a unique pick-up or drop-off point. The last type is usually the so-called feeder line, where flexible transportation service is used to move passengers to another, less accessible service (for instance, communications from rural settlements to an airport). Figure 1 shows a schematic representation of the commented use cases.

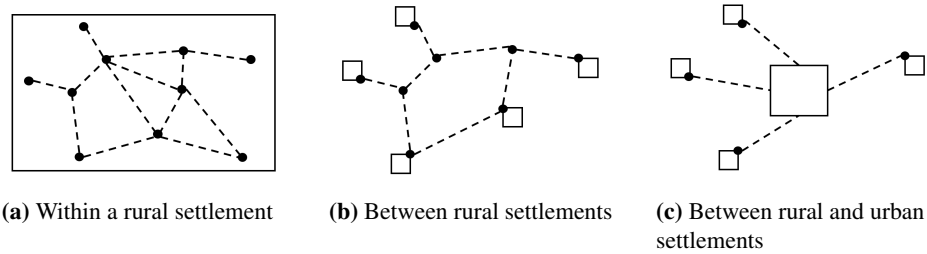


Fig. 1. Observed use cases for rural demand-responsive transportation systems. Boxes indicate rural/urban settlements. Black dots represent stops. Dashed lines represent demand-responsive lines. Pictures (a) and (b) are cases of many-to-many transportation, while (c) represents a many-to-one model

If the customer is required to send a *request* to access transport, then the service is provided *on-demand*. The time between sending a request and the customer's pick up is the *lead time*, and it is used to adapt the fleet operation or *planning* to include such a request. In a stop-based operation, the customer will be assigned a stop from which they will be picked up. On-demand systems can operate based on reservations issued in advance by the users, and in real-time, accepting last-minute bookings. The more complete systems employ a hybrid approach, accepting advanced reservations as well as real-time traveling requests. DRT systems that are not on-demand are also possible. These systems consider current demand or demand predictions for service planning but do not require requests to run.

The period of time for which the DRT service is planned and optimized is referred to as *planning horizon*. The duration of planning horizons is usually a whole day. In addition, the operator may plan for a few hours to adapt to high/low demand periods. According to the influence of the demand data on the service planning, the system will be *fully-*

flexible if routes are planned from scratch according to current demand, or *semi-flexible* if a predetermined plan exists but vehicles are allowed to modify it influenced by demand.

2.2. Modeling and Optimization Techniques

Once the concrete type of DRT system has been chosen, it must be modeled and tested to check its performance and adjust its attributes. We will discuss below the different steps this involves, citing relevant research and their authors' methods. Please be aware that not every paper cited in this section explores rural DRT.

Most rural DRT works are set in a concrete rural settlement or area. In general, the main transportation network (roads, highways) of the area is mirrored thanks to services like OpenStreetMap (<http://openstreetmap.org>) or OpenSourcingRoutingMachine (OSRM, <http://project-osrm.org>) [9]. Ideally, the actual organization of the area, its types of districts, population, or socio-economic reality, among others, should also be considered. Authors in [15] describe a seven-step analysis method for optimizing any transportation system based on reproducing the features of the currently implemented transport service (that would potentially be replaced). Alternatively, some works employ grid-like modelings of the area where the system will run [5].

Demand modeling is also crucial. Passenger demand has two main aspects: (1) frequency and intensity and (2) shape (location of origin-destination pairs). Demand attributes can be extracted from datasets of different transportation modes and extrapolated, as in [13], where taxi data is used. Moreover, real data of pilot DRT services [26,7] can be reproduced when available. However, the most observed technique is the use of synthetic demand data that can be generated statistically [5], based on socio-demographic information [29], via surveys [15,24,9] or generated in a (semi-)random [27] way according to the properties of the reproduced area (population, age, occupation, vehicle ownership). Finally, if traffic intensity data is available, it is useful to include it in the model, although not as relevant for rural areas with respect to city-centered studies since the former tend to have lower intensity.

The operation of the DRT system requires automated planning and scheduling of vehicle services. At the same time, these tasks need information on the time and traveled kilometers that a concrete detour would imply, which makes routing algorithms also necessary. In addition, since it is common to find online systems that accept real-time requests, the computation time for detours and new request insertions must be kept low. The use of multi-modal planning [9] is common to solve the scheduling of vehicle services. Moreover, some simulation platforms, such as MATSim [2] include their own implementations of the algorithms mentioned above. These implementations usually employ (meta)heuristic techniques [29] that optimize vehicle-passenger assignments (insertion heuristics [4], for instance) or vehicle routing in a short computational time. Besides that, other less exploited techniques, such as automated negotiation, could be used to decide assignments from a decentralized perspective [3].

Finally, to observe the system's dynamics and its operation and adjust its attributes, it is necessary to simulate it. This can be performed through mathematical modeling [15] provided detailed data is available. However, a more popular way of achieving this is through multi-agent simulation (MAS). Among the observed choices, we find NetLogo [25], used in [14], the already mentioned MATSim and even custom simulators [20,9].

2.3. Optimization Goals

The main goal of people transportation services is to supply the displacement needs of its users. Ideally, the operation of the service shall be performed by optimizing three factors: (1) the economic viability of the service; (2) the customer's experience (or quality of service); and (3) the sustainability of the service. These three factors are translated into scopes when it comes to transportation research, and thus we can find works that assess one (only operator perspective [18]), or many of them from a multi-objective perspective (passenger and operator perspectives [17]). The optimization of customer experience implies the reduction of passenger travel times, whereas economic viability is ensured by reducing operational costs. Finally, optimizing sustainability requires reducing vehicle traveled kilometers or the total fleet operational time.

The greatest challenge of demand-responsive transportation systems is finding the equilibrium among the factors above to offer a competitively-priced, economically viable, and flexible mobility alternative to private cars and traditional public transportation. For the case of rural DRT, economic viability is especially difficult, taking into account the relatively low demand.

In this section, DRT research has been dissected by reviewing various works. The enumeration of its many configuration options is crucial to plan the correct system according to the characteristics of the area of application. In addition, knowing how authors model and implement their proposals facilitates future research. Coming up, we introduce a proposal for a dynamic DRT system that aids in improving rural mobility.

3. System Proposal

We propose an on-demand, stop-based, many-to-many, and fully-dynamic ride-sharing transportation system to give service to rural areas. A fleet of vehicles provides displacement services with a variable capacity. Each vehicle will follow its own itinerary: the list of stops it will visit during its operation, ordered in time. We assume that users of the system issue travel requests through an application. A travel request indicates the location and time window in a simple manner, such as "Pickup at stop A after 8:30, and dropoff at B by 9:00".

The implementation of our proposal is based on the work in [12]. Our system is managed by a centralized scheduler which allocates each travel request to a vehicle's itinerary. The scheduler has two modes of operation: (1) offline planning of services and (2) online scheduling of incoming travel requests. In offline operation, the scheduler prepares the fleet's itineraries for the following service period (i.e., hours, the following day), finding the optimal allocation of bookings (requests issued in advance). In contrast, during service hours, when the fleet is operating, the scheduler works in online mode, listening to incoming requests and allocating them as they are issued. Figure 2 presents a schematic representation of the scheduler's operation, in which the allocation of a request to an itinerary is referred to as a *trip insertion*.

The scheduler allocates the requests to itineraries such that the system-wide objective function is optimized. Such an objective is the minimization of the fleet's operational time, thus reducing the operational costs of the whole system.

Following, we present the system elements together with their attributes and describe the insertion searching procedure that the scheduler implements.

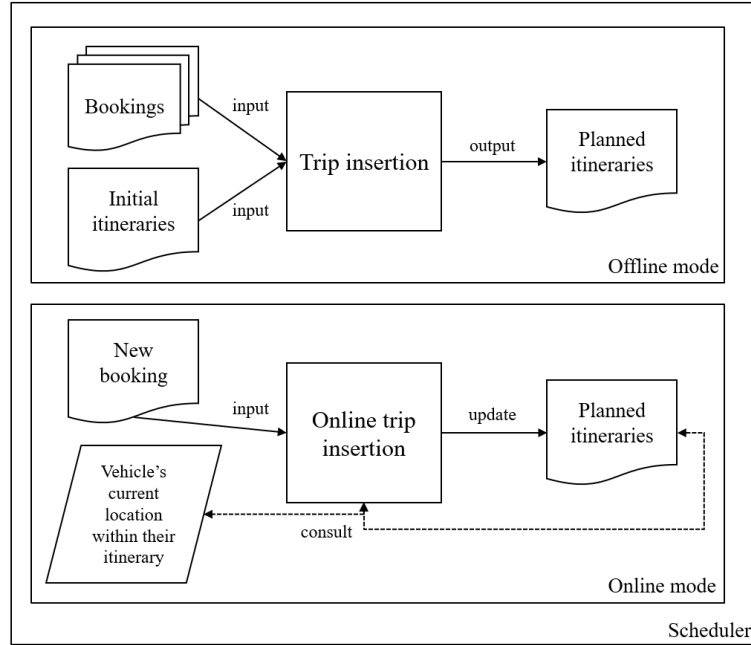


Fig. 2. Operation modes of the proposed transportation system scheduler. Offline refers to static planning of services, whereas online mode indicates the real-time allocation of incoming travel requests

3.1. Definitions

Before describing the request allocation algorithm, it is necessary to define the system's elements. This section briefly enumerates those elements and attributes, giving important notions to understand our implementation. The time units employed in the following formulation are minutes, as these better serve the purposes of our experimentation.

Itineraries. The fleet is managed by the scheduler, a centralized entity with updated information about each vehicle's itinerary, capacity, and location. An itinerary is equivalent to the vehicle it represents. An itinerary is mainly characterized by its stop list, an ordered list of stops that the vehicle will visit, including the time of arrival to and departure from each of them. Even though an itinerary has additional attributes, we underscore that when the text mentions the insertion of an element in an itinerary, it is referring to the itinerary's stop list, as it can be deduced. The attributes of an itinerary I are:

- veh_I : Vehicle represented by itinerary I .
- cap_I : Capacity of veh_I .
- I 's stop list: List of stops of the itinerary; it has at least two stops.
 - S_I^{start} : Stop where veh_I begins its shift, including location and time window.
 - S_I^{end} : Stop where veh_I ends its shift, including location and time window.
- $next_I$: Next stop of veh_I within I 's stop list.

- $cost_I$: Total amount of time that veh_I will spend driving to complete the itinerary.

At the beginning of the operation, an itinerary per fleet vehicle is created. The stop list in those itineraries only contains the stop where its vehicle begins its shift and, subsequently, the stop at which finishes it. As travel requests are assigned to vehicles, the stop list of the vehicle's itinerary is updated, inserting new stops in visiting order. Because of that, the stop list represents the route the vehicle will follow to complete its itinerary.

Trip. The scheduler receives travel requests from the system customers. The request is the explicit petition for displacement. Such a petition describes the displacement in what we call a trip. A trip indicates the need for a certain number of passengers to move from its origin stop to its destination stop. Accepting a request implies that the trip it defines has been inserted in an itinerary, and thus its customers will be serviced. The attributes of a trip t are:

- $npass_t$: Number of passengers traveling as a group on the trip.
- $S_{OR(t)}$: Pickup stop with location and time window.
- $S_{DEST(t)}$: Drop-off stop with location and time window.
- $I_{(t)}$: Itinerary to which the trip is assigned if any. $I_{(t)} \neq \emptyset$ implies $S_{OR(t)}, S_{DEST(t)} \in I_{(t)}$'s stop list.

The time window associated with stop $S_{OR(t)}$ defines the earliest and latest possible times at which the customers can be picked up. Similarly, $S_{DEST(t)}$'s time window defines the earliest and latest drop-off time for the customers. For further clarification on a stop's time window, please refer to the definition of Stops. The wider the time window of a request, the more flexibility the system has to allocate its trip.

Stops. A stop represents a physical location within the transportation service infrastructure where customers can board or lay off a vehicle. In our problem formulation, a stop must be part of a trip or an itinerary. Stops have a time window associated with them. The time window indicates to the scheduler the period of time a stop must be serviced, understanding the service of a stop as the service of the passengers associated with it. When part of a trip, a stop S has the following attributes:

- t_S^{start} : Soonest time at which the stop can be visited by a vehicle. Start of the time window.
- t_S^{end} : Latest time at which the stop can be visited by a vehicle. End of the time window.
- t_S^{serv} : Time employed by a vehicle for passenger pick-up and drop-off at the stop.

In addition, when a stop S is part of the stop list of itinerary I , it has the following attributes, which come in handy to check the feasibility of trip insertions. As a reminder, veh_I indicates the vehicle that follows itinerary I .

- $t_S^{arrival}$: Time at which veh_I arrives to S .
- $t_S^{departure}$: Time at which veh_I departs from S .
- w_S^{serv} : Service window at S , indicating the time taken by passengers boarding or laying off veh_I in S .

- w_S^{wait} : Waiting window at S , during which veh_I waits in S until the departure time.
- $npass_S$: Number of passengers boarded in the veh_I on departure from S .

Given the above attributes, the time window of a stop is defined as follows:

$$t_S^{start} \leq t_S^{arrival}, [w_S^{serv}], [w_S^{wait}], t_S^{departure} \leq t_S^{end}$$

The vehicle visiting a stop can arrive to it at time t_S^{start} as the soonest. Then, the service interval $[w_S^{serv}]$ begins, in which passengers are going on or off the vehicle. Following, the vehicle may wait at the stop for a defined waiting interval $[w_S^{wait}]$. At the end of such a waiting period, the vehicle departs from the stop, which may be at time t_S^{end} at the latest.

The particular arrival and departure times to a stop are determined according to a dispatching strategy. A dispatching strategy defines the use of the so-called slack time, the period of time during which the vehicle does not yet need to leave the stop where it is stationary (represented by w_S^{wait} in our formulation). A general dispatching strategy would be departing the current stop as soon as possible, providing the earliest possible service to those customers of the following stop. In contrast, other strategies force the vehicle to wait at its current stop as much as possible, hoping new requests will be issued and thus having more stationary vehicles to assign them to. For this work, we make use of a hybrid strategy. Vehicles will depart from a stop to ensure the earliest feasible service to the following stop. When the vehicle has slack time, it waits at a stop to maximize the chance of inserting an incoming request.

Insertions. An insertion indicates the feasibility of allocating the trip of a request to a particular itinerary. Moreover, it indicates the positions within the itinerary's stop list where each trip stop will be inserted. The scheduler looks for all feasible insertions of a trip and implements the best one.

Given a trip t , its insertion in an itinerary I implies finding appropriate spots within I 's stop list to visit t 's $S_{OR(t)}$ and $S_{DEST(t)}$. The visit to $S_{DEST(t)}$ must be subsequent (but not necessarily directly after) to that of $S_{OR(t)}$. A trip insertion will always increase the itinerary's duration ($cost_I$).

We define a trip insertion π_{ij} with the following attributes:

- $I(\pi)$: Itinerary in which the trip will be inserted.
- i : Position within I 's stop list where S_{OR} will be inserted.
- j : Position within I 's stop list where S_{DEST} will be inserted.
- Δ_{ij} : Time increment incurred by inserting π in I .

Let us have insertion π_{ij} that allocates trip $t = \langle S_{OR}, S_{DEST} \rangle$ to itinerary $I = [S_I^{start}, S_1, \dots, S_n, S_I^{end}]$. The insertion implies creating two new connections in the itinerary: $(S_{i-1} \rightarrow S_{OR})$ and $(S_{j-1} \rightarrow S_{DEST})$, finally obtaining $I = [S_I^{start}, S_1, \dots, S_{i-1}, S_{OR}, \dots, S_{j-1}, S_{DEST}, \dots, S_n, S_I^{end}]$. Keep in mind that we could have $S_{j-1} = S_{OR}$, as the destination stop could be visited immediately after the origin stop.

The implementation of a trip insertion modifies the planned operation of the vehicle to whose itinerary the trip is allocated. Such a modification may occur during the reservation-based operation of the system or in real-time while the vehicle is already in service. In the former case, the time windows associated with each stop in the vehicle's

itinerary are updated taking into account the visit to the inserted trip stops. In the latter case, time windows are adjusted in the same manner, but the vehicle may need to change its route to reflect the changes in its itinerary's stop list. Such a change of route, however, will not break the time window of any already scheduled stop, as that is taken into account by our scheduling algorithm (see Insertion feasibility checks, under Section 3.2 for further details).

Cost computation & Objective function. As commented on the definition of an itinerary, its cost is equivalent to the time the vehicle it represents spends traveling throughout its list of stops. Given an itinerary I with stop list $= [S_0, S_1, \dots, S_{n-1}, S_n]$, its $cost_I$ would be computed by adding the traveling time between every two consecutive stops in its stop list. Let us assume a function $travelTime(x, y)$, which, given service stops x and y , returns the time taken by a fleet vehicle to travel from x to y in minutes. For an itinerary I with n stops in its stop list, the cost would be computed as shown in Equation 1.

$$cost_I = \sum_{i=0}^{n-1} travelTime(S_i, S_{i+1}), \quad \forall S \in I \quad (1)$$

Given a fleet F of vehicles, the system's objective function is to minimize the total vehicle travel time or distance. This implies direct benefits for both passengers (shorter trips) and the service provider (less operational costs). Such an objective is achieved by the way in which requests are allocated to vehicles. These allocations are done with the insertion search procedure, which works by iteratively finding the best possible insertion for each of the pending requests and implementing it. The search for the best insertion is guided by the cost increment Δ that each feasible insertion may incur to an itinerary's cost $cost_I$. Therefore, the system's objective function can also be described as the minimization of the sum of the cost of each itinerary, as represented by Equation 2.

$$min(\sum cost_I), \quad \forall I \in F \quad (2)$$

3.2. Insertion Search Procedures

An insertion search procedure is the action of finding the best position within an itinerary to allocate a request's trip. In other words, the best moment to visit the trip's origin stop and the same for the destination. Our system implements two insertion search procedures, each for an operation mode (online, offline). Following, both procedures are briefly described, together with the system constraints that ensure the consistency of itineraries as trips are inserted.

Offline insertion search. The offline insertion procedure allocates all bookings to the initially empty itineraries of the fleet. The bookings' trips are inserted one by one, according to issuing time, in the best possible itinerary, i.e., the one that minimizes operational time.

The search works as follows: While there are non-allocated requests, the scheduler selects the next request and extracts its trip t . Given t , with origin stop S_{OR} and destination stop S_{DEST} , we want to obtain all feasible insertions of that trip within all itineraries of the fleet. Algorithm 1 receives the S_{OR} , S_{DEST} , and an itinerary I with N stops. Then,

Algorithm 1: Search for feasible insertions within an itinerary I

Data: S_{OR}, S_{DEST}, I
Result: All feasible insertions of S_{OR}, S_{DEST} in I

```

1  $found \leftarrow []$ ; /* List to store feasible insertions */
2  $n \leftarrow 0$ ; /* Pointer to first stop,  $N =$  number of stops in  $I$  */
3 while  $n < N$  do
4    $R \leftarrow I[n]$ ; /* Select stop in position  $n$  */
5   if  $(R \rightarrow S_{OR})$  is feasible then
6      $i \leftarrow n + 1$ ; /* Position to insert  $S_{OR}$  */
7      $I' \leftarrow I.insert(S_{OR}, i)$ , recalculate time constraints;
8      $m \leftarrow i$ ; /* Pointer to  $S_{OR}$  */
9     while  $m < N$  do
10       $R \leftarrow I[m]$ ;
11      if  $(R \rightarrow S_{DEST})$  is feasible then
12         $j \leftarrow m + 1$ ; /* Position to insert  $S_{DEST}$  */
13         $I'' \leftarrow I'.insert(S_{DEST}, j)$ , recalculate time constraints;
14         $\Delta_{ij} \leftarrow cost_{I''} - cost_I$ ; /* Increase in duration */
15         $found \leftarrow found + (\pi_{ij}, \Delta_{ij})$ ;
16      else
17         $m \leftarrow m + 1$ ; /* Go to next stop */
18      end
19    end
20  else
21     $n \leftarrow n + 1$ ; /* Go to next stop */
22  end
23 end
24 return  $found$ ;

```

it returns all feasible insertions found for trip t in I . This is done for all itineraries of the fleet, and all the returned insertions are ordered according to their time increment Δ . The scheduler then implements the insertion with a lower Δ . The request is rejected if the procedure does not find any feasible insertion.

As it can be seen, Algorithm 1 tries to insert S_{OR} in every possible position within I . Once a feasible position is found for S_{OR} , it is inserted in a copy of I , and the time windows of other stops are updated, thus creating itinerary I' . Then, the process tries to insert S_{DEST} in the position of all stops subsequent to S_{OR} in I' . Once a feasible position is found for S_{DEST} , it is inserted in a copy of I' , and the time windows of other stops are updated, thus creating itinerary I'' . We have found a feasible insertion at this point, so the algorithm computes its time increment (comparing I'' and I 's costs) and stores it before continuing the exploration. Please note that I' and I'' are simply auxiliary itineraries; thus, neither I nor the stops in t are modified by the search algorithm. The described procedure constitutes a complete exploration of the possible insertions, allowing the scheduler to implement the optimal one.

Online insertion search. The online insertion procedure works similarly to the offline one but considers the current position of the vehicles within their itineraries. There-

fore, given a trip t and an itinerary I being considered for its insertion, assuming veh_I is traveling the connection $(R \rightarrow next_I)$, Algorithm 1 only explores positions within $[next_I, S_I^{end}]$ for the insertion of the trip's origin and destination stops.

If the trip's origin stop were to be scheduled in $next_I$'s position, we would have an immediate request, which implies the rerouting of veh_I , changing its following stop from $next_I$ to S_{OR} .

Insertion feasibility checks. For the system to work correctly, all itineraries must be consistent. This consistency is enforced through time and capacity constraints.

Let S be a stop in an itinerary I . Let veh_I be the vehicle represented by itinerary I , with a capacity of cap_I . Let $npass_S$ be the number of passengers on board veh_I on departure from S . The capacity constraint states that: $npass_S \leq cap_I, \forall S \in I$. Simply put, the number of passengers on departure from any of the stops of an itinerary can be, at most, the capacity of the vehicle following such an itinerary.

Concerning time constraints, the system implements the following:

- All passengers must be picked up within the time window specified by their request's start time and the maximum waiting time.
- All passengers must get to their destination before their request's end time.
- All stops must have service windows contained within their arrival and departure.
- All stops must be reached within their time window.

An insertion will be *feasible* if the insertion of its trip in its itinerary does not violate any of the above constraints. The developed insertion search procedure returns only feasible insertions. Because of that, the insertion of a trip in an itinerary will never cause any inconsistencies or constraint violations.

Computational complexity. The presented insertion search procedures perform an exhaustive analysis of every possible position in which to allocate a trip within all the fleet's itineraries. This procedure composes a subproblem of the resolution of the whole DRT service, which will be solved once all travel requests have been dealt with.

Regarding the trip insertion search procedure, its computational complexity depends on the number of stops that the itinerary being explored contains. Such a number of stops, in addition, is generally incremented every time a trip is inserted in the itinerary. This causes the search for trip insertion at the beginning of the operation to be less complex than towards its end. Assuming an itinerary has n stops, the complexity of the search is of $\mathcal{O}(n^2)$, as the algorithm checks each feasible position for the trip's origin stop and, for each of these positions, explores all feasible positions for the destination stop, using two nested loops. In practice, the actual search for an insertion is less costly, as the many restrictions that a feasible insertion has to preserve facilitate early discarding of invalid positions within the stop list.

When it comes to the complexity of solving the scenario, we must take into account that the aforementioned search is performed for every travel request (trip) and every vehicle (itinerary) in the fleet. Thus, the computational complexity of allocating T trips within I itineraries is of $\mathcal{O}(T \times I \times n^2)$.

As it can be understood, the service schedules travel requests iteratively according to their issuance time, following a FIFO logic. This way of operating is mandatory in

the online scheduling of requests, as future demand is unknown. Because of that, the resolution of the proposed DRT service is performed greedily and is sensitive to the order in which requests are fed to the scheduler. To palliate this, improvement procedures could be implemented, which considered global cost optimizations over a solved scenario.

4. Experimental Results

This section tests the proposed system's potential to satisfy rural mobility demand. For that, we defined simulations that reproduce the system's operation over a concrete rural area. Following, the rural area where the simulations are set is described. Then, the results of various simulations are presented, showing the evolution of the overall service quality of the system according to demand intensity and fleet size.



Fig. 3. Rural sub-area chosen for the deployment of the proposed system. The area features many small-to-medium-sized settlements. The northern part of the area shows the city of Valencia, Spain

4.1. Rural Use Case Description

A rural sub-area of the region of Valencia, Spain, was chosen for the deployment of the demand-responsive service. For that, we departed from the existing public interur-

ban bus service of the Valencian Community, which connects many rural settlements between them and with the region's main cities. The dataset³, publicly accessible thanks to the Generalitat Valenciana (<https://linkshortner.net/kkvFj>, accessed on December 15th, 2022), contains information on the different transportation lines, routes and stops the service offered. Specifically, it describes 722 lines with a total of 4562 stops. From those, only the elements lying inside the area shown in Figure 3 were kept. That amounted to 88 lines and 341 stops, shown in Figure 4. Since we propose dynamic DRT, the bus lines effectively disappeared, as now vehicles move freely between the stops scheduled in their itinerary. The existing stops, however, were clustered so that any two stops were at least 500 meters apart. With this, the final distribution of 99 stops that can be seen in Figure 5 (left) is obtained. With fewer stops and longer distances between them, a better representation of interurban displacement is achieved.

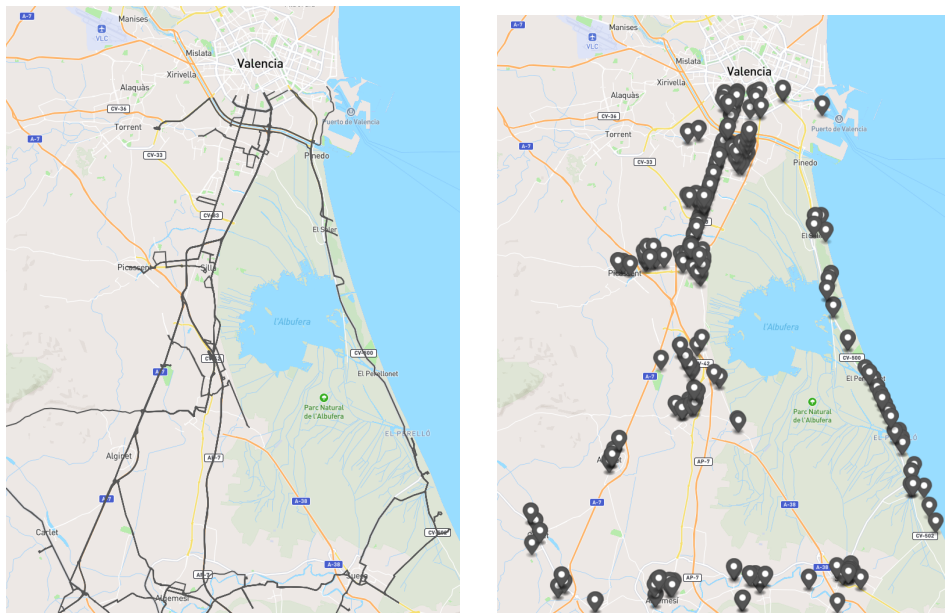


Fig. 4. Bus lines (left) and stops (right) the public interurban bus service defines in the assessed rural area

The deployment area features mainly small-to-medium-sized towns located in rural contexts. It can also be noticed how the urban density increases in the northern part of the area, which is closer to the city of Valencia. Our proposal aims to provide on-demand transportation to citizens of the shown settlements, such as Alginet, Algemesí, Silla, Picassent, and El Saler, to mention a few. Figure 5 (right) shows a close-up in which the location of stops can be better appreciated. Specifically, it shows the town of Sueca and many smaller settlements nearby.

³ <https://dadesobertes.gva.es/va/dataset/gtfs-itineraris-horaris-transport-public-interurba-autobus-comunitat-valenciana>

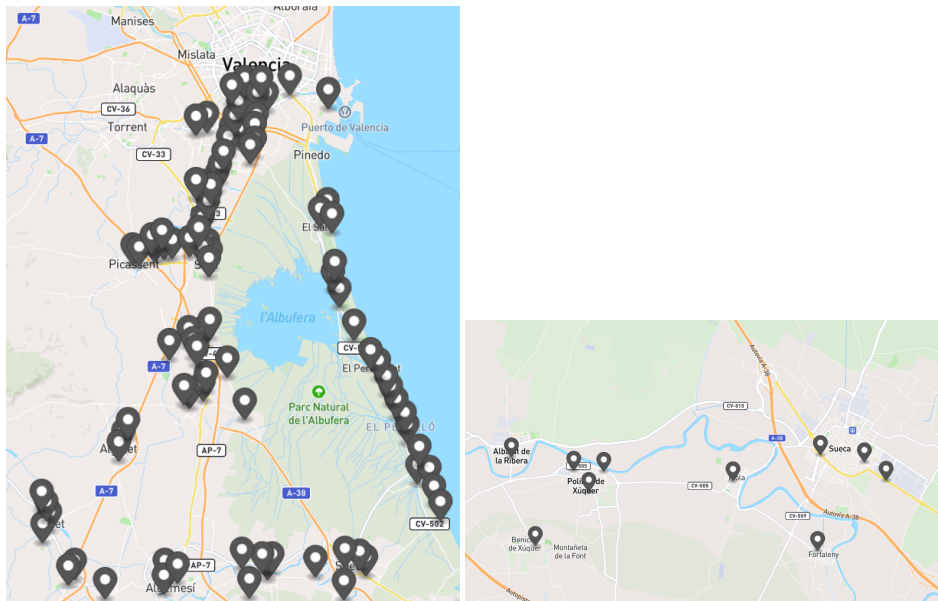


Fig. 5. Final distribution of 99 stops over the chosen deployment area (left). All stops are at least 500 meters apart. The image on the right shows a close-up view of small settlements in the southeastern part of the area, near the town of Sueca

With respect to the displacement demand, the dataset did not provide usage data. To the best of our knowledge, there is no publicly available usage data for interurban displacement within the chosen region. Rural transportation demand has a lower intensity than that of a city, and given the service area, it tends to be widely distributed in space. With that in mind, a synthetic demand generator was employed to feed data to the simulations.

The demand generator receives geolocated population information of the service area to create demand according to it. The more population nearby a stop, the more probable it is to be selected as the trip's origin. The destination stop of the request, however, is chosen randomly among all stops, considering a configurable minimum trip distance. Longer trip distances favor the reproduction of interurban displacements. In addition, each request can have between 1 and 5 passengers with respect to given probabilities (less probable the more people). The demand is uniformly distributed throughout the service hours of the system. The end of a request's time window (the time at which the passengers need to be at their destination) is computed according to a chosen maximum waiting time (at a stop to be serviced) and the direct travel time between origin and destination. The direct travel time is multiplied by a configurable factor. The higher this factor, the wider the time window, and thus the more flexibility the system has to serve the request.

4.2. Service Quality Assessment

The proposed system has been tested through many 14-hour services (07:00 AM to 09:00 PM) simulations with different amounts of vehicles and travel requests. Inspired by the reviewed literature, a fleet of 10 vans, each with a capacity for eight people, was fixed for the first round of experiments. The vans were deployed from a warehouse in Valencia (the northern part of the service area) at 06:00 AM, an hour before the first requests could be scheduled. Similarly, the drivers had to end their shift at the warehouse no later than 10:00 PM.

With regard to the demand, a total number of travel requests was specified and then generated as described above in Section 4.1. The demand is divided into 50% of bookings (scheduled before the system's operation) and another 50% of real-time requests. Each request could have either 1, 2, 3, 4, or 5 passengers with a probability of 0.6, 0.15, 0.125, 0.1, and 0.025, respectively. Finally, a minimum trip distance of 2,000 meters and a maximum waiting time of 15 minutes were chosen. It must be noted that the different probabilities that influence demand generation determine the importance of the subsequent results. For the purposes of demonstrating the proposed algorithm's operation, those probabilities defined above have been used. We remark that the results presented below are dependent on the specific demand generation. Nevertheless, their assessment can give insights to guide future work in this field.

With the fixed fleet of 10 vans, we explored the system's service quality as the number of requests increased. Service quality is defined as the percentage of accepted requests with respect to the total number of requests. In addition, the time that passengers wait for a vehicle to pick them up is included as an additional measurement of service quality. As commented above, for a request to be accepted, their passengers must be picked up before a wait of 15 minutes. Nevertheless, waiting times closer to such a maximum indicate worse passenger experiences. Because of that, our results reflect the average waiting time of all accepted passengers, together with its standard deviation. Table 1 shows our first results. The running time of the most complex simulation was 30 seconds, being executed in a machine running Windows 11 with an Intel Core i7-10750H CPU at 2.60GHz and 16GB of memory.

The system maintained near-perfect service quality in runs with 100 to 300 requests (rows 1 to 5). As it can be seen in the last column, given a particular fleet, the system tries to schedule trips so that all vehicles are employed. Only in the first run, with 100 requests, a vehicle is unused. With 350 requests, the system maintains an acceptable service quality with 84.29% of scheduled requests. From 400 requests on, the service quality decays, lowering to 70% with 450 requests and 62.8% with 500 requests. These last three runs present an unacceptable quality of service ($< 80\%$) based on similar works of the literature. With regards to the average waiting times, results show how these increase proportionally to the number of requests. The standard deviation, however, is kept around 5 minutes throughout all executions. This fact reflects the high variability among each of the individual waiting times, which in turn is motivated by the differences among the generated trips. The obtained average times indicate that most of the passengers are picked up relatively soon after the issuance of their travel requests.

Fleet size. After the initial experimentation, the fleet was varied by adding or subtracting a few vehicles. Once again, the aim was to observe service quality and vehicle usage

Table 1. Service quality evolution with increasing demand and a fixed fleet of 10 vehicles

Requests	req/hour	Vehicles	Capacity	Service quality (%)	Avg. pax wait (min)	Fleet usage
100	~8	10	8	100.00	3.5 ± 5.0	9/10
150	~11	10	8	99.33	4.4 ± 5.2	10/10
200	~15	10	8	99.00	4.3 ± 5.1	10/10
250	~18	10	8	96.00	4.8 ± 5.0	10/10
300	~22	10	8	89.67	5.0 ± 4.9	10/10
350	~25	10	8	84.29	5.5 ± 5.3	10/10
400	~29	10	8	74.75	6.1 ± 5.2	10/10
450	~33	10	8	70.00	6.2 ± 5.1	10/10
500	~36	10	8	62.80	6.5 ± 5.3	10/10

evolution. For these tests, the number of requests increased from 200 to 500 in 50 request intervals. Table 2 presents all the runs. The results indicate that reducing the fleet also reduces the amount of demand the system can appropriately manage, as can be expected. Similarly, with a more significant fleet, the quality of service is preserved above the 70% margin for higher intensities of demand. Even in runs with a more extensive fleet, the system achieves a uniform division of requests among vehicles, employing all of them. The pattern of evolution of passenger waiting times is observed to be the same as in the previous experimentation, having standard deviations approaching 5 minutes across all the tested parameter combinations.

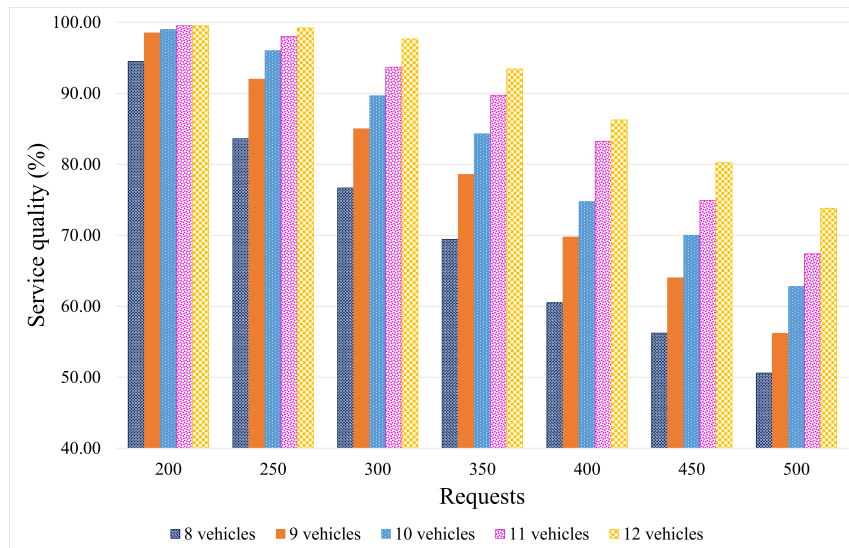


Fig. 6. Visualization of service quality according to various number of requests and fleet sizes

Table 2. Service quality evolution with different fleets ranging from 8 to 12 vehicles and various demand intensities

Requests	req/hour	Vehicles	Capacity	Service quality (%)	Avg. pax wait (min)	Fleet usage
200	~15	8	8	94.50	4.2 ± 5.0	8/8
250	~18	8	8	83.60	5.5 ± 5.1	8/8
300	~22	8	8	76.67	6.3 ± 5.2	8/8
350	~25	8	8	69.43	5.9 ± 5.3	8/8
400	~29	8	8	60.50	6.4 ± 5.1	8/8
450	~33	8	8	56.22	6.5 ± 5.2	8/8
500	~36	8	8	50.60	6.9 ± 5.3	8/8
200	~15	9	8	98.50	4.5 ± 5.3	9/9
250	~18	9	8	92.00	4.8 ± 4.9	9/9
300	~22	9	8	85.00	5.2 ± 4.9	9/9
350	~25	9	8	78.57	5.9 ± 5.3	9/9
400	~29	9	8	69.75	6.2 ± 5.0	9/9
450	~33	9	8	64.00	6.5 ± 5.1	9/9
500	~36	9	8	56.20	6.5 ± 5.2	9/9
200	~15	11	8	99.50	4.2 ± 5.1	11/11
250	~18	11	8	98.00	4.7 ± 5.1	11/11
300	~22	11	8	93.67	4.4 ± 4.9	11/11
350	~25	11	8	89.71	5.1 ± 5.1	11/11
400	~29	11	8	83.25	5.6 ± 5.0	11/11
450	~33	11	8	74.89	6.0 ± 5.0	11/11
500	~36	11	8	67.40	7.0 ± 5.3	11/11
200	~15	12	8	99.50	4.2 ± 5.1	12/12
250	~18	12	8	99.20	4.3 ± 4.9	12/12
300	~22	12	8	97.67	4.3 ± 4.8	12/12
350	~25	12	8	93.43	4.9 ± 5.1	12/12
400	~29	12	8	86.25	5.4 ± 5.0	12/12
450	~33	12	8	80.22	6.1 ± 5.2	12/12
500	~36	12	8	73.80	6.4 ± 5.3	12/12

The graph on Figure 6 visually represents the results of Tables 1 and 2, showing the evolution of the service quality provided by fleets of various vehicles with respect to an increasing number of requests. Table 3 summarizes all results, showing the lower bounds of acceptable service quality found for each combination of demand and fleet size.

Table 3. Lower bound of acceptable service quality found for all combinations of demand intensity and fleet sizes

Requests	req/hour	Vehicles	Capacity	Service quality (%)	Avg. pax wait (min)	Fleet usage
250	~18	8	8	83.60	5.5 ± 5.1	8/8
300	~22	9	8	85.00	5.2 ± 4.9	9/9
350	~25	10	8	84.29	5.5 ± 5.3	10/10
400	~29	11	8	83.25	5.6 ± 5.0	11/11
450	~33	12	8	80.22	6.1 ± 5.2	12/12

Vehicle capacity. The final parameter that was assessed was vehicle capacity. The above simulations were run with fleets of 8 to 12 vehicles but changing their capacity to that of a minibus, ranging from 16 to 22 passengers. The results in terms of quality of service, however, were very similar to what has been presented so far. This indicates that, given the shape of the generated demand, vehicle capacity was not a bottleneck of the system, and rejected requests were motivated by time window incompatibilities and not because of capacity constraints. We must acknowledge, however, that the conclusions drawn from this study of vehicle capacity are only applicable to the specific generated demand. From a general perspective, varying the capacity of fleet vehicles could have a great impact on the system's performance, which is what motivated this final experimentation.

5. Discussion

Given the results summarized in Section 4.2, we can conclude that dynamic DRT is a good fit for the synthetically generated rural mobility demand. The inefficiency of traditional interurban public mobility options in rural contexts comes from the shape of its demand. Vehicles with a high occupancy ratio, scheduled in periodic lines, tend to drive mostly empty, therefore being costly to maintain for public transport providers. The proposed system tackles these problems by ensuring maximum fleet usage, taking advantage of every present vehicle. In addition, this behavior eases the consideration of adding new vehicles to the fleet, as the fleet administrator has the certainty that it will be exploited and thus not a waste of resources.

With regard to the economic viability of the system, having a smaller fleet of smaller vehicles implies lower maintenance and salary expenses. Furthermore, if autonomous mobility becomes feasible in the future, economic expenses would lower even more due to the avoidance of driver salaries. Our experimentation has not explicitly considered the service's environmental impact. Nevertheless, the proposed system has features which

indirectly contribute to a better sustainability. On the one hand, the objective function reduced vehicle travel time which, in turn, would reduce any type of emissions stemming from the fleet. In addition, we assess a reduction of such a fleet, achieving a similar level of service quality while cutting costs. Finally, it is worth mentioning that the environment is better preserved because the fleet makes journeys only when necessary. Moreover, these journeys are more cost-effective due to the higher occupancy of the vehicles.

As seen throughout Section 2, demand-responsive systems present a high number of operation modes and configurable parts. The present work describes one of the many approaches that could work to modernize and improve rural mobility. Ideally, the proposed system would completely replace the inefficient, traditional transportation options. However, in reality, the adoption rate of DRT tends to be low, even more in rural contexts, due to the necessity to explicit a travel request. The easiest methods to do so consist of smartphone applications and call centers, being the former generally harder to manage for the older population. Because of that, the deployment of a demand-responsive system would initially complement the current mobility options providing, for instance, connection to the most stranded settlements with the main means of public transportation.

Finally, we want to assess the lack of publicly available demand data, which hardens the research on rural mobility. In the context of rural DRT, this issue is aggravated by the lack of rural-specific or low-demand datasets. There are a small number of DRT pilot projects, and among them, an even smaller number share the collected data. Still, the data that can be found about pilot projects is very dependent on the specific area and the socio-demographic context where the pilot took place. To deal with data shortage, synthetic data generation is often employed, basing generation on population, age, occupation, and any other kind of survey that characterizes the potential users of the system.

6. Conclusion

In this paper, DRT has been characterized, together with the challenges rural mobility presents for the implementation of efficient modes of public transportation that satisfy the population. A DRT system has been proposed to match the rural mobility demand and provide such a quality service. The system has been described in depth, implemented, and tested by means of simulations. A rural area in the region of Valencia, Spain, has been chosen for the deployment of the system. The mobility demand, in terms of travel requests, has been generated with a synthetic demand generator according to the population of the deployment area and a series of configurable parameters. The research results prove the potential that DRT holds to develop dynamic, reliable, and cost-effective public transportation in the rural context. This research contributes with a system proposal and its validation to the field of rural mobility, which has a general lack of innovation when it comes to displacement proposals.

In terms of future research, we observe two paths. On the one hand, the proposed system can be further improved. Different system configurations must be assessed to find the best match for the deployment area. In addition, the parameters of the proposed system could also be fine-tuned through more experimentation. To further improve results, global optimization techniques can be implemented in order to further optimize the obtained itineraries. For instance, considering request exchange among vehicles could decrease global costs. Finally, we would like to include transfer operations as an option for

the scheduler to allocate requests. These operations have the potential to simplify the fleet operation, cutting costs. On the other hand, regarding experimentation, it would be interesting to assess the impact of different levels of demand dynamism, tighter request time windows, or different dispatching strategies, to mention a few. Finally, simulation results could be enhanced by considering factors such as vehicle autonomy or strategic agent behavior.

As closing remarks, we want to state that there is a need for specific investigations on the successful implementation of DRT. To bridge such a gap, researchers must go beyond service quality to focus on the adoption rate and usage of the system. For instance, we believe in the potential pricing policies that could both attract new users to the system and, in addition, influence how they use it to improve the overall quality of service.

Acknowledgments. This work is partially supported by grant PID2021-123673OB-C31 funded by MCIN/AEI/ 10.13039/501100011033 and by “ERDF A way of making Europe”. Pasqual Martí is supported by grant ACIF/2021/259 funded by the “Conselleria de Innovación, Universidades, Ciencia y Sociedad Digital de la Generalitat Valenciana”. Jaume Jordán is supported by grant IJC2020-045683-I funded by MCIN/AEI/ 10.13039/501100011033 and by “European Union NextGenerationEU/PRTR”.

References

1. Anburuvel, A., Perera, W., Randeniya, R.: A demand responsive public transport for a spatially scattered population in a developing country. *Case Studies on Transport Policy* 10(1), 187–197 (2022)
2. Balmer, M., Rieser, M., Meister, K., Charypar, D., Lefebvre, N., Nagel, K.: Matsim-t: Architecture and simulation times. In: *Multi-agent systems for traffic and transportation engineering*, pp. 57–78. IGI Global (2009)
3. Bertelle, C., Nabaa, M., Olivier, D., Tranouez, P.: A decentralised approach for the transportation on demand problem. In: *From System Complexity to Emergent Properties*, pp. 281–289. Springer (2009)
4. Bischoff, J., Maciejewski, M.: Proactive empty vehicle rebalancing for demand responsive transport services. *Procedia Computer Science* 170, 739–744 (2020)
5. Calabrò, G., Le Pira, M., Giuffrida, N., Inturri, G., Ignaccolo, M., Correia, G.: Fixed-route vs demand-responsive transport feeder services: An exploratory study using an agent-based model. *J. of Advanced Transportation* 2022 (2022)
6. Coutinho, F.M., van Oort, N., Christoforou, Z., Alonso-González, M.J., Cats, O., Hoogendoorn, S.: Impacts of replacing a fixed public transport line by a demand responsive transport system: Case study of a rural area in amsterdam. *Research in Transportation Economics* 83, 100910 (2020)
7. Coutinho, F.M., van Oort, N., Christoforou, Z., Alonso-González, M.J., Cats, O., Hoogendoorn, S.: Impacts of replacing a fixed public transport line by a demand responsive transport system: Case study of a rural area in amsterdam. *Research in Transportation Economics* 83, 100910 (2020)
8. Currie, G., Fournier, N.: Why most drt/micro-transits fail—what the survivors tell us about progress. *Research in Transportation Economics* 83, 100895 (2020)
9. Dyckov, S., Persson, J.A., Lorig, F., Davidsson, P.: Potential benefits of demand responsive transport in rural areas: A simulation study in lolland, denmark. *Sustainability* 14(6) (2022)

10. Enoch, M., Potter, S., Parkhurst, G., Smith, M.: Why do demand responsive transport systems fail? In: Transportation Research Board 85th Annual Meeting. Washington DC, USA (22-26 Jan 2006)
11. Ho, S.C., Szeto, W., Kuo, Y.H., Leung, J.M., Petering, M., Tou, T.W.: A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological* 111, 395–421 (2018)
12. Horn, M.E.: Fleet scheduling and dispatching for demand-responsive passenger services. *Transportation Research Part C: Emerging Technologies* 10(1), 35–63 (2002)
13. Hyland, M., Mahmassani, H.S.: Operational benefits and challenges of shared-ride automated mobility-on-demand services. *Transportation Research Part A: Policy and Practice* 134, 251–270 (2020)
14. Inturri, G., Giuffrida, N., Ignaccolo, M., Le Pira, M., Pluchino, A., Rapisarda, A.: Testing Demand Responsive Shared Transport Services via Agent-Based Simulations, pp. 313–320. Springer International Publishing, Cham (2018)
15. Lakatos, A., Tóth, J., Mándoki, P.: Demand responsive transport service of ‘dead-end villages’ in interurban traffic. *Sustainability* 12(9) (2020)
16. Li, Y., Qian, Y., Li, Q., Li, L.: Evaluation of smart city construction and optimization of city brand model under neural networks. *Computer Science and Information Systems* 20(2), 573–593 (2023)
17. Liyanage, S., Dia, H.: An agent-based simulation approach for evaluating the performance of on-demand bus services. *Sustainability* 12(10) (2020)
18. Marković, N., Kim, M.E., Kim, E., Milinković, S.: A threshold policy for dispatching vehicles in demand-responsive transit systems. *Promet - Traffic & Transportation* 31(4), 387–395 (Aug 2019)
19. Martí, P., Jordán, J., Julian, V.: Demand-responsive mobility for rural areas: A review. In: González-Briones, A., Almeida, A., Fernandez, A., El Bolock, A., Durães, D., Jordán, J., Lopes, F. (eds.) *Highlights in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. The PAAMS Collection*. pp. 129–140. Springer International Publishing, Cham (2022)
20. Palanca, J., Terrasa, A., Carrascosa, C., Julián, V.: Simfleet: a new transport fleet simulator based on mas. In: *International Conference on Practical Applications of Agents and Multi-Agent Systems*. pp. 257–264. Springer (2019)
21. Roh, C.G., Kim, J.: What are more efficient transportation services in a rural area? a case study in yangsan city, south korea. *International journal of environmental research and public health* 19(18), 11263 (2022)
22. Ryley, T.J., A. Stanley, P., P. Enoch, M., M. Zanni, A., A. Quddus, M.: Investigating the contribution of demand responsive transport to a sustainable local public transport system. *Research in Transportation Economics* 48, 364–372 (2014)
23. Schasché, S.E., Sposato, R.G., Hampl, N.: The dilemma of demand-responsive transport services in rural areas: Conflicting expectations and weak user acceptance. *Transport Policy* 126, 43–54 (2022)
24. Schlüter, J., Bossert, A., Rössy, P., Kersting, M.: Impact assessment of autonomous demand responsive transport as a link between urban and rural areas. *Research in Transportation Business & Management* 39, 100613 (2021)
25. Tisue, S., Wilensky, U.: Netlogo: A simple environment for modeling complexity. In: *Int. conference on complex systems*. vol. 21, pp. 16–21. Boston, MA (2004)
26. Vallée, S., Oulamara, A., Cherif-Khettaf, W.R.: Maximizing the number of served requests in an online shared transport system by solving a dynamic darp. In: *Computational Logistics*. pp. 64–78. Springer International Publishing, Cham (2017)
27. van Engelen, M., Cats, O., Post, H., Aardal, K.: Enhancing flexible transport services with demand-anticipatory insertion heuristics. *Transportation Research Part E: Logistics and Transportation Review* 110, 110–121 (2018)

28. Vansteenwegen, P., Melis, L., Aktaş, D., Montenegro, B.D.G., Vieira, F.S., Sörensen, K.: A survey on demand-responsive public bus systems. *Transportation Research Part C: Emerging Technologies* 137, 103573 (2022)
29. Viergutz, K., Schmidt, C.: Demand responsive - vs. conventional public transportation: A mat-sim study about the rural town of colditz, germany. *Procedia Computer Science* 151, 69–76 (2019)
30. Wang, C., Quddus, M., Enoch, M., Ryley, T., Davison, L.: Exploring the propensity to travel by demand responsive transport in the rural area of lincolnshire in england. *Case Studies on Transport Policy* 3(2), 129–136 (2015)
31. Yao, B., Liu, S., Wang, L.: Using machine learning approach to construct the people flow tracking system for smart cities. *Computer Science and Information Systems* 20(2), 679–700 (2023)

Pasqual Martí is a Ph.D. student and researcher at the Valencian Research Institute for Artificial Intelligence (VRAIN) of the Universitat Politècnica de València. He began his career in research through a scholarship, participating in 2 R&D projects related to Multi-Agent Systems and Urban Fleets. From such work, he has published several articles in indexed journals and international conferences.

Jaume Jordán is a postdoctoral researcher at the Valencian Research Institute for Artificial Intelligence (VRAIN) of the Universitat Politècnica de València. He finished his Ph.D. in 2017 with a grade of Outstanding Cum Laude and international mention. He has participated in more than 10 R&D projects (1 as principal investigator). He has more than 40 works published in relevant international conferences and JCR journals.

Vicente Julian holds the position of Full Professor of Computer Science at the Universitat Politècnica de València (UPV) where he has taught since 1996. He is a member of the Valencian Research Institute for Artificial Intelligence (VRAIN), and the Coordinator of the Computer Science Ph.D. Program at the UPV. 4 international projects, 2 international excellence networks, 25 Spanish projects, and 4 technology transfer projects have covered his research on Artificial Intelligence. He has more than 300 works published in journals with outstanding JCR positions or in relevant conference proceedings, and supervised 10 Ph.D. Thesis.

Received: January 15, 2023; Accepted: October 25, 2023.

How to Fairly and Efficiently Assign Tasks in Individually Rational Agents' Coalitions? Models and Fairness Measures

Marin Lujak^{1,*}, Alessio Salvatore³, Alberto Fernández¹, Stefano Giordani², and Kendal Cousy^{1,2}

¹ CETINIA, University Rey Juan Carlos, Madrid, Spain
{marin.lujak, alberto.fernandez, kendal.cousy}@urjc.es

² Dipartimento di Ingegneria dell'Impresa, University of Rome "Tor Vergata", Rome, Italy
stefano.giordani@uniroma2.it

³ Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti", Consiglio Nazionale delle Ricerche,
Rome, Italy
alessio.salvatore@iasi.cnr.it

Abstract. An individually rational agent will participate in a multi-agent coalition if the participation, given available information and knowledge, brings a payoff that is at least as high as the one achieved by not participating. Since agents' performance and skills may vary from task to task, the decisions about individual agent-task assignment will determine the overall performance of the coalition. Maximising the efficiency of the one-on-one assignment of tasks to agents corresponds to the conventional linear sum assignment problem, which considers efficiency as the sum of the costs or benefits of individual agent-task assignments obtained by the coalition as a whole. This approach may be unfair since it does not explicitly consider fairness and, thus, is unsuitable for individually rational agents' coalitions. In this paper, we propose two new assignment models that balance efficiency and fairness in task assignment and study the utilitarian, egalitarian, and Nash social welfare for task assignment in individually rational agents' coalitions. Since fairness is a relatively abstract term that can be difficult to quantify, we propose three new fairness measures based on equity and equality and use them to compare the newly proposed models. Through functional examples, we show that a reasonable trade-off between efficiency and fairness in task assignment is possible through the use of the proposed models.

Keywords: Task Assignment, Multi-Agent Systems, Fairness, Efficiency, Resource Allocation, Multi-Agent Coordination

1. Introduction

An individually rational agent operates within a decision-making context, driven by self-interest to ensure that it attains a payoff or utility that is at least as favourable as its best alternative, including the option of not participating. When a coalition is formed by individually rational agents, they share a common objective, even though the agents themselves may possess distinct, potentially conflicting interests. They collaborate by pooling

* Corresponding author

their capabilities and/or resources to efficiently carry out designated tasks, often achieving synergistic outcomes that surpass what they could accomplish individually. In this context, agents strive to achieve superior performance through collective action. Instances of such coalitions can be observed in various domains, including emergency services (e.g., [26]), agricultural cooperatives (e.g., [28]), taxi and ride-sharing services (e.g., [5, 27]), as well as smart grids (e.g., [36]), among others.

Of our concern in this paper is the one-on-one agent-task assignment in a coalition composed of individually rational agents. We assume a centralised decision-making process (or an algorithm) that is in charge of deciding which agent is assigned to each task. From a utilitarian point of view, an optimal solution would be the assignment that produces the lowest overall cost (or the highest benefit) for the coalition as a whole. However, the most efficient solution for the overall system may create large differences among the agents in terms of their individually assigned costs (we refer to this as an “unfair” assignment). The perception of an unfair task assignment solution may motivate unsatisfied agents to leave the coalition, putting the survivability of the same at risk. Thus, assignment decisions should be made based on minimising overall assignment costs and considering social welfare and fairness.

The linear-sum assignment problem, related to the topic of this paper, is a largely studied generally computationally easy problem that maximizes the efficiency of a multi-agent system without considering fairness. Exact solutions for this problem can be produced efficiently even for problem instances with a very large number of agents and tasks. However, to the best of our knowledge, related work on balancing fairness and efficiency in task assignment is scarce. Therefore, in this work, we explore the means of balancing the overall cost and fairness in task assignment in agent coalitions. These two aspects are generally opposed, i.e., solution approaches focusing on cost minimisation are likely to produce unfair assignments for some agents, while fair assignments may be far from the minimum cost solution for the coalition as a whole. In this paper, we study the trade-off between these two requirements and focus on finding task assignment solutions that are as fair as possible while not overly penalising the overall coalition’s assignment cost. This implies finding efficient and fair assignments considering the distribution of individual costs among coalition members.

The main contributions of this paper are twofold. First, we propose three new fairness measures for a multi-agent system composed of self-concerned individually rational agents: Egalitarian Fairness Measure (EFM), Relative All-to-all Fairness measure (RAF), and Overall Relative Opportunity Cost Fairness (OROCF) measure. Then, we present two new one-on-one task assignment models that maximise the social welfare of the system while balancing efficiency and fairness: an envy-free utilitarian model that uses the utilitarian social welfare function while constraining the differences in the costs between agents, and the Nash model that optimises the Nash product of assigned tasks’ benefits or costs of individual agents composing the system. We choose Nash social welfare due to its structure (being a product of costs) that explicitly balances efficiency and fairness.

The rest of the paper is organised as follows. In Section 2, we give an overview of the state of the art. In Section 3, we give motivation for this work and define the general problem of one-on-one task assignment. We propose new equality and equity fairness measures in Section 4 and introduce a linearised model for the calculation of one of the more computationally complex fairness measures. The two new proposed mathematical

models for efficient and fair task assignment are presented in Section 5. Section 6 presents simple functional tests and discusses how the presented models differ based on the proposed fairness measures. Following these functional tests, in Section 7, we compile the results of many experiments to have a complete overview of the strengths and weaknesses of each model. In Section 8, we conclude the paper by giving an overview of the results and discuss the potential of the new proposed models and fairness measures to make a fair and efficient task assignment. We also give lines of future work to improve the proposed assignment models and fairness measures.

2. State of the Art

The assignment of resources, chores, or tasks in a multi-agent coalition may vary when defining fairness and efficiency depending on agents' mutual inter-dependencies and their relation with the coalition's objectives (e.g., [8, 11, 29]).

In this paper, we study balancing fairness and efficiency in the allocation of indivisible goods (resources, chores, or tasks) (e.g., [12]), and, more specifically, in the one-on-one assignment of tasks to agents in a cooperative multi-agent coalition composed of individually rational agents.

Cooperative decision making considers working toward a shared goal even though its ownership is not shared [33], as opposed to collaborative decision making, which considers a goal that is shared and owned by all agents in a coalition. Thus, cooperative decision making results are generally differentially beneficial to different agents [30], while collaboration is generally about equally sharing efforts, costs, and benefits.

Cooperative multi-agent task allocation problem was studied in, e.g., [18, 23, 27]. This problem has many different real-world applications where fairness can be a challenge. For example, in spatial crowdsourcing [42], there is a need to minimise the payoff difference among workers while maximising the average worker payoff. Similarly, in ride-share platforms, it was shown in [32] that, during high-demand hours, lacking any consideration of fairness and seeking only an optimal number of trips could lead to increased societal biases in the choice of the clients. This problem is relevant for many other applications including manufacturing and scheduling, network routing and the fair and efficient exploitation of Earth Observation Satellites, among others (e.g., [11]).

Various fairness measures exist for different contexts, e.g., machine learning (e.g., [14]), neural networks (e.g., [34]) and algorithm development (e.g., [20]). Fairness is studied as well in other contexts, like the multi-winner voting problem and recommender systems (e.g., [40]), but also in decision making (e.g., [37]). The importance of the individual perception of fairness within a system to keep individual satisfaction high is emphasised in [38]. In the Machine Learning context, the potential contradiction between individual and group fairness is studied in [6]. Some works study more generally the concepts of distributive justice, equality and equity (e.g., [13]).

The two most well-known fairness measures for the allocation of indivisible goods include max-min and proportional allocation. The max-min fairness aims to maximize the utility of the agent who contributes the least to the overall utility of the system. In other words, it ensures that the agent with the smallest contribution still receives as much utility as possible, while proportional fairness dictates that each agent should receive at least one-nth of the utility they would have obtained if they were the sole recipient of

the goods. Max-min fairness is generalised in the case of resource allocation for systems with different resource types in [17] while max-min fairness, proportional fairness and balanced fairness are compared in the setting of a communication network of processor-sharing queues in [7]. Additionally, in resource allocation, there can be agents desiring some tasks (resources) more than others, or there can even be agents desiring tasks given to other agents, creating envy in the system (e.g., [11]). Envy-freeness criterion implies that an allocation should leave no agent envious of the other (e.g., [9]). However, it is not always enough to achieve envy-freeness for a fair solution (e.g., [2, 22]).

Balancing fairness and efficiency in divisible resource sharing was studied in, e.g., [1], while the related work on the competitive counterpart of cooperative systems usually studies finding a Pareto-optimal and fair allocation of indivisible items aiming at maximising (computationally expensive) Nash welfare (e.g., [4, 19, 41]). Proving the existence of such an allocation for an arbitrary number of agents is still an open problem [3].

Various social welfare concepts play an important role in balancing between efficiency and fairness. Their modelling and importance in enhancing the quality of task allocation are studied in [11]. In this work, we study egalitarianism and utilitarianism in this regard. Egalitarianism is a trend of thought in political philosophy that favours equality among the individuals composing the coalition no matter what their circumstances are (e.g., [16]). Utilitarianism, on the other hand, is a theory of morality that advocates actions that maximise happiness or well-being for all individuals while opposing the actions that cause unhappiness or harm. When directed toward making social and economic decisions, a utilitarian philosophy aims at the improvement of the coalition as a whole (e.g., [31]).

The Nash social welfare combines efficiency and fairness considerations. This function, or variants of it, are studied in literature considering, e.g., fairness in the ambulance location problem [21], and in allocating indivisible goods [10]. The multi-agent resource allocation problem considering Nash social welfare (the product of the utilities of the individual agents) is studied in [35].

This paper is an extended version of [15], where we previously studied the problem of balancing efficiency and fairness in linear-sum one-on-one task assignment. To the best of our knowledge, most of the related works treat efficiency in terms of proportionality in competitive multi-agent systems and propose computationally expensive models suitable for instances with a relatively small number of agents and tasks. In this paper, we propose two scalable computationally efficient models for task assignment in cooperative multi-agent systems, Nash model and envy-free utilitarian model, both with quality of solution guarantees. The proposed models balance efficiency in terms of overall system cost and the proposed fairness measures.

3. Motivation and problem definition

Most of the state-of-the-art literature on task assignment generally focuses on the efficiency of the assignment in terms of cost minimisation or benefit maximisation and does not consider fairness in the process, thus optimising only the system's overall general assignment cost or benefit (e.g., time, distance, monetary value, etc.). It is noteworthy that, from an optimisation standpoint, the task of maximising the overall benefit of multi-agent task assignments can be effectively converted into a cost-minimisation problem by simply inverting the values associated with agent-task assignment benefits.

The overall cost-minimisation approach in multi-agent task assignment is equivalent to optimising utilitarian social welfare function, a concept from welfare economy that sums the utility of each individual to obtain society’s overall welfare (see, e.g., [11, 39]). All agents are treated the same, regardless of their initial level of utility or cost distribution among the tasks. This strategy is admissible in the case of a single decision maker, but might be unacceptable when multiple self-concerned and individually rational agents must mutually decide on the assignment of tasks.

Let us introduce a simple example showing how unfair a task assignment optimising a utilitarian social welfare function can be. Let us consider 3 self-concerned, individually rational agents (a_1, a_2, a_3) that need to be assigned in a one-on-one manner to a set of 3 tasks (k_1, k_2, k_3) and vice versa. The cost matrix containing the assignment costs for these agents and tasks is shown in Table 1a.

Table 1. Example of a cost matrix and different one-on-one task assignment solutions with minimum overall cost (in bold)

(a) Cost Matrix	(b) Solution s_1	(c) Solution s_2	(d) Solution s_3																																																																
<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td></td><td>k_1</td><td>k_2</td><td>k_3</td></tr> <tr><td>a_1</td><td>50</td><td>60</td><td>70</td></tr> <tr><td>a_2</td><td>30</td><td>40</td><td>50</td></tr> <tr><td>a_3</td><td>10</td><td>50</td><td>30</td></tr> </table>		k_1	k_2	k_3	a_1	50	60	70	a_2	30	40	50	a_3	10	50	30	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td></td><td>k_1</td><td>k_2</td><td>k_3</td></tr> <tr><td>a_1</td><td>50</td><td>60</td><td>70</td></tr> <tr><td>a_2</td><td>30</td><td>40</td><td>50</td></tr> <tr><td>a_3</td><td>10</td><td>50</td><td>30</td></tr> </table>		k_1	k_2	k_3	a_1	50	60	70	a_2	30	40	50	a_3	10	50	30	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td></td><td>k_1</td><td>k_2</td><td>k_3</td></tr> <tr><td>a_1</td><td>50</td><td>60</td><td>70</td></tr> <tr><td>a_2</td><td>30</td><td>40</td><td>50</td></tr> <tr><td>a_3</td><td>10</td><td>50</td><td>30</td></tr> </table>		k_1	k_2	k_3	a_1	50	60	70	a_2	30	40	50	a_3	10	50	30	<table border="1" style="border-collapse: collapse; width: 100px; height: 100px;"> <tr><td></td><td>k_1</td><td>k_2</td><td>k_3</td></tr> <tr><td>a_1</td><td>50</td><td>60</td><td>70</td></tr> <tr><td>a_2</td><td>30</td><td>40</td><td>50</td></tr> <tr><td>a_3</td><td>10</td><td>50</td><td>30</td></tr> </table>		k_1	k_2	k_3	a_1	50	60	70	a_2	30	40	50	a_3	10	50	30
	k_1	k_2	k_3																																																																
a_1	50	60	70																																																																
a_2	30	40	50																																																																
a_3	10	50	30																																																																
	k_1	k_2	k_3																																																																
a_1	50	60	70																																																																
a_2	30	40	50																																																																
a_3	10	50	30																																																																
	k_1	k_2	k_3																																																																
a_1	50	60	70																																																																
a_2	30	40	50																																																																
a_3	10	50	30																																																																
	k_1	k_2	k_3																																																																
a_1	50	60	70																																																																
a_2	30	40	50																																																																
a_3	10	50	30																																																																

By applying the conventional (linear-sum) task assignment model (i.e., the utilitarian social welfare model) that minimises the overall assignment cost of the system without considering fairness (see, e.g., [24, 27]), we might get the assignments (called solution s_1) marked in bold in Table 1b where agent a_1 is assigned to task k_2 , agent a_2 is assigned to task k_1 and agent a_3 to task k_3 . The overall (minimum) assignment cost found by this model is 120. However, if we focus on its cost distribution over individual agents, we see large discrepancies. Indeed, the cost of agent a_1 is 60, while the cost of a_2 (and a_3) is only 30. Thus, a_1 is charged twice more than a_2 (and a_3). In Table 1d (i.e., solution s_3), this difference is even larger resulting in a 7 times larger cost of the worst-off concerning the best-off agent. Generally, an upper bound on the difference in the assignment cost is the maximum value in a given cost matrix. In centralised systems, where agents are owned and controlled by a single decision-maker, this would not cause any problems. However, in the case of decentralised systems composed of self-concerned and individually rational agents, such an unfair solution might result in the worst-off agents leaving the system due to the lack of fairness in the solution.

Table 1c shows a fairer solution (that we name s_2) where the costs of the agents are as close as possible, thus minimising the envy of agents. This is an ideal situation regarding fairness in this case where all agents are assigned tasks of similar costs. Notice that, here we did not sacrifice efficiency to achieve balanced individual costs. In case of repetitive task allocations, the assignments can be altered to further facilitate balance in the accumulated assignment costs.

Problem definition. Given are a set of agents $a \in A$ and a set of tasks $k \in K$ that form a weighted complete bipartite graph $G = (A \cup K, E)$ with edge set $E = A \times K$ and with given edge weights c_{ak} on each edge $(a, k) \in E$, where c_{ak} is the cost of assigning task k to agent a , for all $a \in A$ and $k \in K$. W.l.o.g, we assume that the cardinality of the two sets is equal, i.e., $|A| = |K|$. In the case of unequal cardinality, we add a sufficient number of dummy vertices and assume that $c_{ak} = 0$ where $a \in A$ or $k \in K$ are dummy vertices. Moreover, for simplicity, we assume that agents are indexed from 1 to $|A|$, i.e., $A = \{1, \dots, |A|\}$. The objective is to assign agents $a \in A$ to tasks $k \in K$ in a one-on-one manner and, therefore, find a perfect matching among vertices in A and vertices in K considering both assignment efficiency and fairness. An edge (a, k) is matched if its (two) extreme vertices $a \in A$ and $k \in K$ are mutually matched, and matching is perfect if every vertex in A is matched (assigned) to exactly one vertex in K , and vice versa. The following is the mathematical formulation of these constraints.

$$\sum_{k \in K} x_{ak} = 1, \forall a \in A \quad (1) \quad \sum_{a \in A} x_{ak} = 1, \forall k \in K \quad (2)$$

$$x_{ak} \in \{0, 1\}, \forall a \in A, \forall k \in K \quad (3)$$

where x_{ak} is a binary decision variable equal to 1 if agent $a \in A$ is assigned to task $k \in K$, and zero otherwise. Constraints (1) and (2) assure that there is one-on-one assignment for each agent $a \in A$ and task $k \in K$, respectively. Constraints (3) fix the ranges of the decision variables.

4. Proposed fairness measures

In this section, we propose three new fairness measures for quantifying the balance between fairness and efficiency in task assignment from the egalitarian and equity points of view. All the fairness measures are fractions ranging between 0 and 1. We avoid the division by 0 in some extreme cases by adding a very small number ϵ (e.g., $\epsilon = 1e^{-10}$) to both the numerator and the denominator of these fractions.

The proposed fairness measures should be computed only for non-dummy vertices in the bipartite graph that represents the agents and tasks to ensure that these measures can still reach either the value of 0 or 1.

4.1. Egalitarian Fairness Measure (EFM)

Egalitarian Fairness Measure (EFM) focuses on the assignment cost faced by the worst-off agent (i.e., the agent with the highest assignment cost in a given feasible solution). Given the assignments x_{ak}^{sol} , with $a \in A$ and $k \in K$, of a feasible solution sol , EFM is computed as follows:

$$EFM(sol) = \frac{c_{max} - c_{sol}^{wo} + \epsilon}{c_{max} - c_{min}^{wo} + \epsilon} \quad (4)$$

where $c_{max} = \max_{a \in A, k \in K} \{c_{ak}\}$ is the maximum value in the cost matrix, $c_{sol}^{wo} = \max_{a \in A} \{\sum_{k \in K} c_{ak} x_{ak}^{sol}\}$ is the cost paid by the worst-off agent in the given solution,

and c_{min}^{wo} is the minimum (or the preferred) cost that the same agent could pay for task assignment. In particular, c_{min}^{wo} is the optimal solution of the given mathematical problem:

$$c_{min}^{wo} = \min \lambda \quad (5)$$

s.t. (1)–(3) and

$$\sum_{k \in K} c_{ak} x_{ak} \leq \lambda, \forall a \in A \quad (6)$$

$$\lambda \geq 0 \quad (7)$$

where constraints (6) impose the upper limit on the cost (λ) paid by the worst-off agent, and Constraints (7) fix the range of the additional variable λ . When the worst-off assigned cost c_{sol}^{wo} is equal to c_{max} , $EFM(sol)$ will equal zero (ignoring ϵ). On the other hand, when c_{sol}^{wo} is equal to c_{min}^{wo} , $EFM(sol)$ will equal one; moreover, this also occurs when there exists an agent $a \in A$ such that $c_{ak} = c_{max}$, for all $k \in K$.

For the cost matrix given in Table 1a, where $c_{max} = 70$ and $c_{min}^{wo} = 50$, we calculate the $EFM(sol)$ for each solution reported in Tables 1b–1d. All the solutions reported in Table 1, have a minimum overall assignment cost equal to 120, while the values of c_{sol}^{wo} are $c_{s_1}^{wo} = 60$, $c_{s_2}^{wo} = 50$, and $c_{s_3}^{wo} = 70$, for the solutions reported in Table 1b, Table 1c, and Table 1d, respectively. $EFM(sol)$ value for these solutions are: $EFM(s_1) = \frac{70-60}{70-50} = 0.5$, $EFM(s_2) = \frac{70-50}{70-50} = 1$, and $EFM(s_3) = \frac{70-70}{70-50} = 0$.

According to the EFM measure definition, solution s_2 is the fairest one. Note that the increase in EFM value in solution s_2 corresponds to a distribution of the costs that leaves the worst-off agent better off than in s_1 , and that solution s_3 leaves the worst-off agent with the worst possible (highest) cost. Note that, generally, there may be multiple such distributions.

4.2. Relative All-to-all Fairness (RAF) measure

Relative All-to-all Fairness (RAF) measure evaluates fairness at a society level by taking into account every agent's assignment in comparison with the others. The measure considers the squared differences of the assignment costs of each agent concerning the costs of all the others, as seen in Equation (8).

$$w_{sol} = \sum_{a \in A} \sum_{a' \in A | a' > a} \left(\sum_{k \in K} c_{ak} x_{ak}^{sol} - c_{a'k} x_{a'k}^{sol} \right)^2 \quad (8)$$

Then, relative all-to-all fairness is computed as follows:

$$RAF(sol) = \frac{w_{max} - w_{sol} + \epsilon}{w_{max} - w_{min} + \epsilon}, \quad (9)$$

where w_{max} and w_{min} represent the maximum and the minimum value for the RAF fairness measure that should be calculated a priori for a specific data set. They are modelled as follows:

$$w_{min} = \min \sum_{a \in A} \sum_{a' \in A | a' > a} \left(\sum_{k \in K} c_{ak} x_{ak} - c_{a'k} x_{a'k} \right)^2, \quad (10)$$

and

$$w_{max} = \max \sum_{a \in A} \sum_{a' \in A | a' > a} \left(\sum_{k \in K} c_{ak} x_{ak} - c_{a'k} x_{a'k} \right)^2, \quad (11)$$

both s.t. (1)–(3).

For the cost matrix given in Table 1a, the two components of RAF that are independent of the assignment solution are $w_{max} = 5400$ and $w_{min} = 0$, related to solutions s_{max} , with $x_{13}^{s_{max}} = x_{22}^{s_{max}} = x_{31}^{s_{max}} = 1$, and s_{min} , with $x_{11}^{s_{min}} = x_{23}^{s_{min}} = x_{32}^{s_{min}} = 1$, respectively. The values w_{sol} for the solutions reported in Table 1b, 1c and 1d are $w_{s_1} = 1800$, $w_{s_2} = 600$, and $w_{s_3} = 5400$, respectively. Related *RAF* values are: $RAF(s_1) = \frac{5400-1800}{5400-0} = 0.67$, $RAF(s_2) = \frac{5400-600}{5400-0} = 0.89$, and $RAF(s_3) = \frac{5400-5400}{5400-0} = 0$.

Also according to the RAF measure, solution s_2 is the fairest one and the order of the three solutions is the same as for EFM. This is not surprising as both measures evaluate the equality of a solution. However, s_2 is not the absolute fairest solution which, concerning this indicator, is $x_{11} = x_{23} = x_{32} = 1$ where all the agents pay the same cost; in this case, the RAF value is equal to 1. This is also not surprising as this particular measure considers not only the worst-off agent, but all of them, therefore making it less likely that one of the solutions with minimum cost also has the highest fairness value.

Linearisation of the RAF measure. It is computationally expensive to find the maximum value w_{max} and the minimum value w_{min} for the RAF fairness measure since their models (10) and (11), respectively, are composed of quadratic terms. This is especially the case in larger problem instances.

To fix this issue, in this section, we simplify the RAF measure by making a linear model that will be easier to solve. We first modify the RAF formula by exchanging the quadratic expressions with the absolute ones and then linearise the latter ones.

RAF measure with absolute values. We present in the following the proposed modified models, replacing the quadratic terms with absolute ones, subject to the same constraints as before.

$$w'_{min} = \min \sum_{a \in A} \sum_{a' \in A | a' > a} \left| \sum_{k \in K} c_{ak} x_{ak} - c_{a'k} x_{a'k} \right|, \quad (12)$$

and

$$w'_{max} = \max \sum_{a \in A} \sum_{a' \in A | a' > a} \left| \sum_{k \in K} c_{ak} x_{ak} - c_{a'k} x_{a'k} \right|, \quad (13)$$

both s.t. (1)–(3).

Minimising the RAF measure with linearised absolute values. In order to determine the optimal value of w'_{min} in (12) subject to (1)–(3), we linearise objective function (12) by adding new constraints and continuous free variables $r_{aa'}$, for all $a, a' \in A$, with $a < a'$, as follows.

$$w'_{min} = \min \sum_{a \in A} \sum_{a' \in A | a' > a} r_{aa'}, \quad (14)$$

s.t. (1)–(3), and

$$r_{aa'} \geq \sum_{k \in K} [c_{ak}x_{ak} - c_{a'k}x_{a'k}], \forall a, a' \in A, \text{ with } a < a', \quad (15)$$

$$r_{aa'} \geq \sum_{k \in K} [c_{a'k}x_{a'k} - c_{ak}x_{ak}], \forall a, a' \in A, \text{ with } a < a', \quad (16)$$

Maximising the RAF measure. In order to determine the optimal value of w'_{max} in (13) subject to (1)–(3), next, we add other constraints and binary variables $y_{aa'}, \forall a, a' \in A$, with $a < a'$, in our problem. We let $M = 2 \cdot \max_{a \in A, k \in K} \{c_{ak}\}$ be a parameter that is introduced for constraints (20) and (21) to select the largest of the two possible terms. We present next the modified problem.

$$w'_{max} = \max \sum_{a \in A} \sum_{a' \in A | a' > a} r_{aa'}, \quad (17)$$

s.t. (1)–(3), and

$$r_{aa'} \geq \sum_{k \in K} [c_{ak}x_{ak} - c_{a'k}x_{a'k}], \forall a, a' \in A \text{ with } a < a', \quad (18)$$

$$r_{aa'} \geq \sum_{k \in K} [c_{a'k}x_{a'k} - c_{ak}x_{ak}], \forall a, a' \in A \text{ with } a < a', \quad (19)$$

$$r_{aa'} \leq \sum_{k \in K} [c_{ak}x_{ak} - c_{a'k}x_{a'k}] + M \times y_{aa'}, \forall a, a' \in A \text{ with } a < a', \quad (20)$$

$$r_{aa'} \leq \sum_{k \in K} [c_{a'k}x_{a'k} - c_{ak}x_{ak}] + M \times (1 - y_{aa'}), \forall a, a' \in A \text{ with } a < a', \quad (21)$$

$$y_{aa'} \in \{0, 1\}, \forall a, a' \in A \text{ with } a < a', \quad (22)$$

4.3. Overall Relative Opportunity Cost Fairness (OROCF)

Overall Relative Opportunity Cost Fairness (OROCF) focuses on evaluating equity among the agents by taking into account the missed opportunities in terms of the assignment cost for each agent. The opportunity cost (e.g., [25]) is the concept in the microeconomics of lost benefit that would have been derived by an agent from an option not chosen. As the reference value, we consider a task of the minimum cost and normalise the difference in the cost value between the assigned task and the best-off task (the task with minimum cost) over the amplitude of costs for each agent, as seen in Equation (23).

$$y_{sol} = \sum_{a \in A} \frac{\sum_{k \in K} c_{ak} x_{ak}^{sol} - \min_{k \in K} \{c_{ak}\} + \epsilon}{\max_{k \in K} \{c_{ak}\} - \min_{k \in K} \{c_{ak}\} + \epsilon} \quad (23)$$

$$OROCF(sol) = \frac{y_{max} - y_{sol} + \epsilon}{y_{max} - y_{min} + \epsilon}, \quad (24)$$

where y_{max} , y_{min} represents the maximum and the minimum value of Equation (23) given Constraints (1)–(3).

The values y_{sol} for the solutions reported in Tables 1b, 1c and 1d are $y_{s_1} = 1$, $y_{s_2} = 1$, and $y_{s_3} = 1.5$, respectively. For the cost matrix given in Table 1a, the two components of OROCF that are independent of the assignment solution are $y_{max} = 2$, for $x_{13}^{s_{max}} = x_{21}^{s_{max}} = x_{32}^{s_{max}} = 1$, and $y_{min} = 1$, for $x_{11}^{s_{min}} = x_{22}^{s_{min}} = x_{33}^{s_{min}} = 1$. Related OROCF values are: $OROCF(s_1) = \frac{2-1}{2-1} = 1$, $OROCF(s_2) = \frac{2-1}{2-1} = 1$, and $OROCF(s_3) = \frac{2-1.5}{2-1} = 0.5$.

Note that OROCF value is the highest both for s_1 and s_2 , meaning that these solutions offer the lowest highest opportunity cost for the sum of all agents. The reader can verify that the solution $x_{11} = x_{23} = x_{32} = 1$ would be the worst choice for agents a_2 and a_3 and would give a value of OROCF equal to 0.

5. Proposed models considering fairness and efficiency

In this section, we propose new models that mitigate the equity issues posed by the classical linear-sum assignment model (e.g., [8]) and achieve a solution that is as fair as possible while sacrificing as little as possible the overall system's efficiency.

5.1. Nash Model for task assignment

The proposed Nash Model is inspired by the Nash social welfare function, a well-studied social welfare function in which the goal is to maximize the product of the utility functions of the agents composing the system. The proposed model is given next:

$$\min \prod_{a \in A} \sum_{k \in K} c_{ak} x_{ak} \quad (25)$$

s.t. (1)–(3). Since Eq. (25) is a nonlinear objective function, solving the above problem is computationally expensive. We propose next its linearised equivalent, which is possible due to the one-on-one assignment constraints (1)–(3).

$$\max \sum_{a \in A} \sum_{k \in K} \log(M - c_{ak}) x_{ak} \quad (26)$$

s.t. (1)–(3), where $M > \max_{k \in K, a \in A} \{c_{ak}\}$.

5.2. Envy-free utilitarian model for task assignment

We propose the envy-free utilitarian model that focuses both on efficiency and fairness. We introduce the fairness variable f_u to ensure that all the costs for each agent are inside a certain interval that shrinks as f_u becomes smaller. The model is defined as follows:

$$\min \alpha f_u + (1 - \alpha) \frac{\sum_{k \in K} \sum_{a \in A} c_{ak} x_{ak}}{|A|} \tag{27}$$

s.t. (1)–(3), and

$$\sum_{k \in K} c_{ak} x_{ak} - \frac{\sum_{k \in K} \sum_{a \in A} c_{ak} x_{ak}}{|A|} \leq f_u, \forall a \in A \tag{28}$$

$$f_u \geq 0, \tag{29}$$

where fairness weight α in objective function (27) ranges between 0 and 1 and is used to weigh the fairness f_u and the average cost paid by the agents' coalition $\sum_{k \in K} \sum_{a \in A} c_{ak} x_{ak} / |A|$; when $\alpha = 0$, the model only considers the cost without considering fairness and vice versa for its value equal to 1. Constraints (28) guarantee that, for each agent, the difference between the cost of its assigned task and the average of the costs of the assigned tasks for all the agents is less than the value f_u . Constraint (29) fixes the range of variable f_u .

6. Functional tests

To evaluate the performance of the Nash model and the Envy-free Utilitarian model, we randomly generate three cost matrices (Table 2) with costs ranging from 1 to 1000. The models were solved for each matrix using IBM ILOG CPLEX Optimization Studio 20.0.1.

Table 2. Example cost matrices

(a) Functional test 1

	k_1	k_2	k_3
a_1	382	816	366
a_2	846	544	175
a_3	578	824	526

(b) Functional test 2

	k_1	k_2	k_3
a_1	450	895	358
a_2	856	233	449
a_3	890	672	976

(c) Functional test 3

	k_1	k_2	k_3
a_1	683	170	699
a_2	943	364	894
a_3	557	741	127

To compare the efficiency of the models presented in Section 5, we calculate the following normalised efficiency indicator (*Eff*):

$$Eff(sol) = \frac{z_{max} - z_{sol} + \epsilon}{z_{max} - z_{min} + \epsilon} \tag{30}$$

where $z_{sol} = \sum_{k \in K} \sum_{a \in A} c_{ak} x_{ak}^{sol}$ with x_{ak}^{sol} being the solution returned by the considered model. The values z_{max} and z_{min} are, respectively, the maximum and the minimum

values of $\sum_{k \in K} \sum_{a \in A} c_{ak} x_{ak}$ given Constraints (1)–(3). Table 3 shows the results of our experiments.

Table 3. Results and Comparison

Functional test 1				
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>
Nash	0.91	1	1	1
Envy-free ($\alpha = 0$)	1	0.07	0	0.68
Envy-free ($\alpha \geq 0.5$)	0.91	1	1	1
Functional test 2				
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>
Nash	0.93	1	0.91	1
Envy-free ($\alpha = 0$)	1	0.28	0.17	0.92
Envy-free ($\alpha = 0.5$)	0.93	1	0.91	1
Envy-free ($\alpha \geq 0.9$)	0	0	1	0
Functional test 3				
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>
Nash	0.93	1	0.73	1
Envy-free ($\alpha = 0$)	1	0	0	0.99
Envy-free ($\alpha = 0.5$)	0.93	1	0.73	1
Envy-free ($\alpha = 0.7$)	0.59	0.94	0.93	0.64
Envy-free ($\alpha \geq 0.9$)	0.05	0.19	1	0.06

The case when $\alpha = 0$ corresponds to the case when we are optimising the global cost only (utilitarian social welfare function). We get very low values of fairness for this case according to our prior assumptions. It is interesting to notice similarities when we set the α value to 0.5. Indeed, in that case, the Envy-free Utilitarian model and the Nash model have the same behaviour and give us the same solutions. These solutions for $\alpha = 0.5$ are ideal for the fairness indicators *EFM* and *OROCF* in our three tests, while *RAF* also increases significantly. Moreover, the efficiency (*Eff*) is greater than 0.9. Equality and equity can be improved without a significant decrease in efficiency. We notice in tests 2 and 3 that, generally, the higher the value of α , the lower the overall system's efficiency. This shows that striving for too much equality can be highly detrimental to the system's efficiency and even equity. The results for the cost matrix in Table 1a also support this claim in case $\alpha = 1$. Here, allocation $x_{11} = x_{23} = x_{32} = 1$ is an egalitarian allocation that decreases efficiency and equity simultaneously since agents a_2 and a_3 are allocated to their worst-off tasks and the overall allocation cost is 150 instead of the minimum cost of 120.

7. Simulation experiments

In this section, we perform simulation experiments to evaluate how the proposed models behave from the scalability standpoint.

For these experiments, 20 cost matrices $C = \{c_{ak} | a \in A, k \in K\}$ where $|A| = |K|$ of size 5, 10 and 20 were created, with each cost c_{ak} randomly obtaining a value from 1 to

Table 4. Results depending on problem size

Size 5					
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>	<i>z</i>
Nash	1.00	0.95	0.77	0.99	1402
Envy-free ($\alpha = 0$)	1.00	0.89	0.75	1.00	1399
Envy-free ($\alpha = 0.25$)	0.99	0.96	0.77	0.99	1405
Envy-free ($\alpha = 0.5$)	0.94	1.00	0.82	0.94	1520
Envy-free ($\alpha = 0.75$)	0.73	0.89	0.91	0.73	1913
Envy-free ($\alpha = 1$)	0.41	0.62	0.94	0.41	2513
$z_{min} = 1339; z_{max} = 3408$					
Size 10					
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>	<i>z</i>
Nash	1.00	0.95	0.87	1.00	1439
Envy-free ($\alpha = 0$)	1.00	0.93	0.87	1.00	1438
Envy-free ($\alpha = 0.25$)	1.00	0.99	0.88	1.00	1466
Envy-free ($\alpha = 0.5$)	0.97	1.00	0.89	0.98	1626
Envy-free ($\alpha = 0.75$)	0.87	0.98	0.96	0.88	2330
Envy-free ($\alpha = 1$)	0.47	0.59	0.99	0.48	5235
$z_{min} = 1438; z_{max} = 8602$					
Size 20					
Model	<i>Eff</i>	<i>EFM</i>	<i>RAF</i>	<i>OROCF</i>	<i>z</i>
Nash	1.00	0.95	0.94	1.00	1522
Envy-free ($\alpha = 0$)	1.00	0.95	0.94	1.00	1522
Envy-free ($\alpha = 0.25$)	1.00	1.00	0.96	1.00	1610
Envy-free ($\alpha = 0.5$)	0.98	1.00	0.95	0.99	1789
Envy-free ($\alpha = 0.75$)	0.92	0.99	0.98	0.92	2924
Envy-free ($\alpha = 1$)	0.46	0.52	0.97	0.46	10687
$z_{min} = 1522; z_{max} = 18473$					

1000 following a uniform law of probability. This results in a total of 60 instances that we test in the experiments in Table 4. For the envy-free utilitarian model, α takes the values of 0 (the classical utilitarian model), 0.25, 0.5, 0.75 and 1 (considering only fairness f_u).

Table 4 lists the average results for size $n = 5, 10, 20$, respectively, where z is the average value of the sum of the assignment costs. It is to be noted that RAF this time is calculated using the RAF minimisation and maximisation models from section 4.2. In addition, the z_{min} and z_{max} values from the efficiency formula *Eff* (30) are given just after the sub-table for each problem size.

Experiments results considering only lower bound. Table 5 presents the values of our four fairness measures as a ratio of their absolute values to their lower bounds. The experiments in this section were conducted on randomly generated instances of varying sizes: 5, 10, 20, 50, and 100, with 20 instances for each size.

This allows for a comparison of the models based solely on their relative performance to the lower bound, rather than taking into account both bounds. In this way, the difference between the models is clearer compared to the results of Table 4.

The fairness measures are calculated as follows:

Table 5. Results depending on problem size with only lower bounds

Size 5				
Model	Eff'	EFM'	RAF'	$OROCF'$
Nash	1.00	1.05	2.17	1.02
Envy-free ($\alpha = 0$)	1.00	1.09	2.25	1.02
Envy-free ($\alpha = 0.25$)	1.01	1.04	2.15	1.04
Envy-free ($\alpha = 0.5$)	1.11	1.00	1.90	1.67
Envy-free ($\alpha = 0.75$)	1.42	1.04	1.41	3.52
Envy-free ($\alpha = 1$)	1.92	1.30	1.21	12.72

Size 10				
Model	Eff'	EFM'	RAF'	$OROCF'$
Nash	1.00	1.11	1.93	1.00
Envy-free ($\alpha = 0$)	1.00	1.15	1.96	1.01
Envy-free ($\alpha = 0.25$)	1.02	1.02	1.86	1.05
Envy-free ($\alpha = 0.5$)	1.13	1.00	1.76	1.35
Envy-free ($\alpha = 0.75$)	1.63	1.03	1.32	2.94
Envy-free ($\alpha = 1$)	3.87	2.00	1.10	10.37

Size 20				
Model	Eff'	EFM'	RAF'	$OROCF'$
Nash	1.00	1.22	1.63	1.00
Envy-free ($\alpha = 0$)	1.00	1.22	1.63	1.00
Envy-free ($\alpha = 0.25$)	1.06	1.01	1.49	1.15
Envy-free ($\alpha = 0.5$)	1.17	1.00	1.52	1.51
Envy-free ($\alpha = 0.75$)	1.91	1.04	1.23	3.58
Envy-free ($\alpha = 1$)	7.23	3.02	1.39	19.89

Size 50				
Model	Eff'	EFM'	RAF'	$OROCF'$
Nash	1.00	1.24	1.15	1.02
Envy-free ($\alpha = 0$)	1.00	1.30	1.16	1.02
Envy-free ($\alpha = 0.25$)	1.03	1.01	1.09	1.09
Envy-free ($\alpha = 0.5$)	1.33	1.00	1.23	2.05
Envy-free ($\alpha = 0.75$)	2.22	1.05	1.03	4.80
Envy-free ($\alpha = 1.00$)	30.50	11.57	1.03	86.73

Size 100				
Model	Eff'	EFM'	RAF'	$OROCF'$
Nash	1.00	1.25	1.07	1.00
Envy-free ($\alpha = 0$)	1.00	1.15	1.08	1.00
Envy-free ($\alpha = 0.25$)	1.01	1.00	1.05	1.04
Envy-free ($\alpha = 0.5$)	1.53	1.00	1.48	2.53
Envy-free ($\alpha = 0.75$)	2.41	1.48	1.98	3.77
Envy-free ($\alpha = 1.00$)	56.67	16.84	1.02	156.86

- $EFM' = c_{sol}^{wo}/c_{min}^{wo}$
- $RAF' = w_{sol}/w_{min}$
- $OROCF' = y_{sol}/y_{min}$
- $Eff' = z_{sol}/z_{min}$,

where the used terminology is explained in Section 4. The values for these metrics are in the range $[1.0, +\infty]$, with 1.0 indicating the best performance. The table shows the average for the values obtained on 20 samples.

7.1. Interpretation of the results

The different models offer a good choice to pick from depending on the desired goal. In the case that the system is aiming at giving the agents a similar cost, then using the envy-free utilitarian model and increasing the value of α will considerably help. If however, the system is aiming at an all-round and fast solution for allocating the tasks with both equity and equality taken into account, then the Nash model will work very well. Indeed, if we compare the Nash model to the Utilitarian model ($\alpha = 0$), which gives us the best solution for the system taken as a whole in terms of the cost paid by the agents, for each size we can notice that all the fairness measures are better, closer to 1.0. What is even more noticeable is that while being fairer overall, the efficiency of the Nash model is really good, with the global cost of the solutions being roughly just as low as the Utilitarian model ($\alpha = 0$).

As we can see in Table 5, the Envy-Free Utilitarian Model (EFUM) decreases its *efficiency* as α increases, since a higher α gives more importance to the fairness (envy-free) part. If $\alpha = 0$, only the utilitarian part is considered, thus the optimal utilitarian solution is obtained ($Eff' = 1$). The Nash model turns out to be optimal, from the utilitarian point of view, in most experiments (on average, $Eff' = 1$).

The *egalitarian fairness metric (EFM)* measures how much the cost of the worst-off agent is kept as low as possible in a given agent-task assignment. Total envy-freeness ($\alpha = 1$) is not the best option because it is possible to obtain higher differences among assigned costs (lower envy-freeness) even at a lower cost for the worst-off agent. Conversely, the optimal utilitarian solution ($\alpha = 0$) does not consider the worst-off agent cost at all and obtains a much higher EFM' value. The best result from the egalitarian (EFM) point of view is obtained by the EFUM with α values near 0.5, i.e. with a balance between the fairness and utilitarian perspectives. The Nash model behaves quite similarly to the total envy-freeness ($\alpha = 1$) since it obtains the optimal solution in most of our experiments.

Regarding *RAF*, while it seems that increasing α for the EFUM gives better results with smaller problem sizes, it is not consistent with size 20 ($\alpha = 0.5$ and $\alpha = 1$), size 50 ($\alpha = 0.5$) and size 100 ($\alpha = 0.5$ and $\alpha = 0.75$). The Nash model, as mentioned above, behaves very similarly to EFUM with $\alpha = 0$, thus it obtains bad results from the *RAF* measure point of view.

With regards to *OROCF* measure, EFUM is better for smaller α , i.e. in case we give more importance to efficiency than to envy-freeness. We assume this is because *OROCF* measures the loss against the most efficient solution from each local viewpoint. Since the Nash model obtains the optimal solution in most of the cases, its *OROCF'* value can be also very good (1).

7.2. Run time comparison

Table 6 shows CPLEX solving times (in seconds) we obtain when comparing the initial quadratic version of *RAF* and the absolute version of *RAF*, for both minimising and maximising the related evaluation functions. The values shown are an average of 20 samples for each size $n = 5, 8, 10$.

Table 6. RAF run time comparison depending on size

size	Quadratic RAF run time (s)		Absolute RAF run time (s)	
	Min	Max	Min	Max
5	0.066	0.113	0.035	0.110
8	1.329	2.575	0.075	2.323
10	354.104	456.932	0.368	57.183

We can see a huge improvement (reduction) in the time it takes for the solver to find the solution for both the minimisation and the maximisation of the modified RAF measure with absolute terms in comparison with the one with quadratic terms. This difference is more noticeable with matrices of bigger size.

Now, looking at the average solving time for our models on the graphs of Fig. 1, tested with four values of alpha for the envy-free model which are 0.25, 0.5, 0.75 and 1.0, we can see that using the Utilitarian and Nash models we get the fastest results, with roughly half a second of running time even for problems of size 200.

When solving the Envy-free model, we observe an exponential increase in the solving time, with problems not being solved optimally after one hour for even small sizes of the problem when alpha gets close to 1.

Stars in yellow in the graph indicate problem sizes for which CPLEX was unable to find an optimal solution within an hour. In such instances, we used the relative Mixed Integer Programming (MIP) gap to assess the proximity of the found solution to the optimal solution. The MIP gap serves as an indicator of the proximity of the current solution to the optimal solution. It is defined as the difference between the upper bound (representing the best-known upper bound on the objective value obtained thus far) and the lower bound (representing the best-known lower bound on the objective value obtained thus far), divided by the absolute value of the upper bound. This measure, expressed as a percentage, provides a relative value, offering insights into the potential further reduction in the objective value of the model after the prescribed one-hour run-time limit.

Here, when $\alpha = 0.75$, the gap is 52% on average for problems of size 100, and it is 76% on average for problems of size 200. When $\alpha = 1.0$, the gap is 100% even for problems of size 20, 50, 100 and 200. An MIP gap of 100% is a clear indication that the solver has not yet been able to find a feasible solution within the defined constraints and within the prescribed one-hour run-time limit.

8. Conclusions

In this paper, we studied the means of balancing efficiency and fairness in one-on-one agent-task assignment in agent coalitions composed of individually rational agents. Here, an agent decides to collaborate with other agents only if it brings an individual benefit that is at least as good as when not collaborating. In this regard, we studied the utilitarian, egalitarian and Nash social welfare, the concepts from economics and philosophy that may be applied in such multi-agent coalitions to tackle this issue. Since quantitative fairness measures for task assignment are scarce and/or missing, we proposed three new fairness measures: egalitarian fairness measure (EFM), relative all-to-all fairness measure (RAF), and overall relative opportunity cost fairness (OROCF) measure. Moreover, to improve

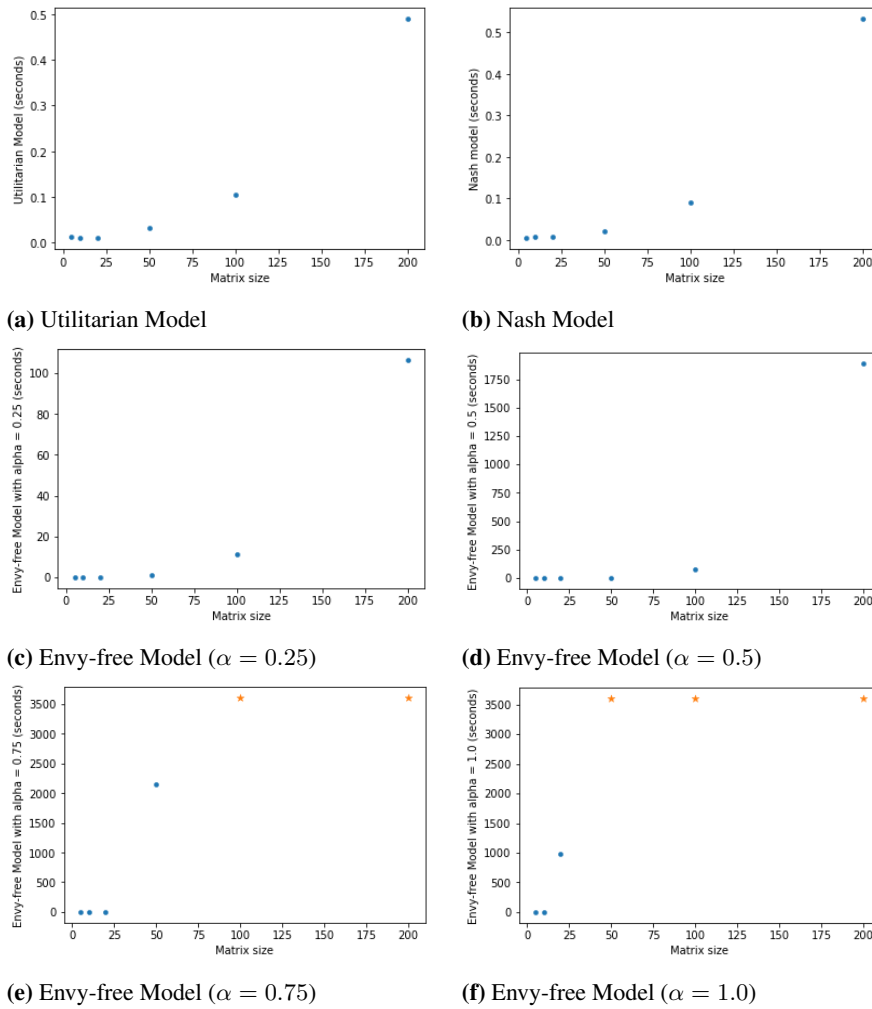


Fig. 1. Run time of the models depending on problem size

the performance of the conventional task assignment model, we proposed the Nash model for the one-on-one task assignment that minimises the product of the costs of each agent considering one-on-one assignment constraints and the envy-free utilitarian model that combines the envy-freeness concept, equity and the utilitarian social welfare measure. The performed simulation experiments show that by using our newly proposed models, we can achieve better fairness in terms of the proposed measures with little sacrifice in the overall efficiency and that the envy-free utilitarian model can be adjusted depending on the need for fairness in a coalition.

The potential impact of the proposed models and fairness measures in real-world applications is substantial. To implement the proposed system effectively, it is imperative to establish an a priori agreement or contract defining the efficiency and fairness measures for the multi-agent coalition.

In the domain of ride-sharing and delivery services, companies like Uber and Lyft can employ these models to optimize efficiency while ensuring fairness among their drivers. This approach can lead to improved driver satisfaction and retention rates. Additionally, in labour markets and the gig economy, these concepts can be harnessed to allocate freelance and short-term work equitably through digital platforms. This benefits both workers and employers by fostering a more engaged and content workforce. Furthermore, our proposed models and fairness measures may be invaluable in supply chain management, disaster response coordination, environmental conservation efforts, healthcare settings, academic research collaborations, and smart grid management. These models excel at promoting computationally efficient task allocation that strikes a balance between efficiency and fairness, potentially resulting in enhanced productivity, coordination, and cooperation across a diverse range of industries and sectors.

In future work, we plan to further study fairness measures, particularly the one encompassing both equality and equity to better support decision-making in collaborative and cooperative open societies where agents can enter and leave the system at any time based on their momentary interest. Moreover, we will focus on the three-index assignment problem where each agent requires a tool to perform a task. The assignment here is also performed in a one-on-one manner. Similarly, crafting a multi-objective model which considers equality, equity and fairness for such a problem is a challenge worth facing henceforth due to the importance of its impact in versatile real-world applications. These include emergency services, agriculture fleet task coordination, delivery services, waste management, construction and infrastructure projects, utility maintenance, and home healthcare, among others.

Acknowledgments. This work has been partially supported by grants PID2021-123673OB-C32 and TED2021-131295B-C33 funded by MCIN/AEI/10.13039/501100011033 and by “ERDF A way of making Europe” and the “European Union NextGenerationEU / PRTR”, respectively; as well as the AGROBOTIX Project 2022-REGING-95993 funded by the IMPULSO own program of the University Rey Juan Carlos and AGROBOTS Project funded by the Community of Madrid, Spain.

References

1. Airiau, S., Aziz, H., Caragiannis, I., Kruger, J., Lang, J., Peters, D.: Portioning using ordinal preferences: Fairness and efficiency. *Artificial Intelligence* **314**, 103809 (2023)

2. Alkan, A., Demange, G., Gale, D.: Fair allocation of indivisible goods and criteria of justice. *Econometrica: Journal of the Econ. Soc.* **59**(4) (1991)
3. Aziz, H., Caragiannis, I., Igarashi, A., Walsh, T.: Fair allocation of indivisible goods and chores. *Autonomous Agents and Multi-Agent Systems* **36**, 1–21 (2022)
4. Aziz, H., Li, B., Moulin, H., Wu, X.: Algorithmic fair allocation of indivisible items: A survey and new questions. *ACM SIGecom Exchanges* **20**(1), 24–40 (2022)
5. Billhardt, H., Fernandez, A., Lujak, M., Ossowski, S., Julian, V., De Paz, J.F., Hernandez, J.Z.: Coordinating open fleets. a taxi assignment example. *AI Communications* **30**(1), 37–52 (2017)
6. Binns, R.: On the apparent conflict between individual and group fairness. In: *Proc. of the ACM FAccT 2020*. pp. 514–524 (2020)
7. Bonald, T., Massoulié, L., Proutiere, A., Virtamo, J.: A queueing analysis of max-min fairness, proportional fairness and balanced fairness. *Queueing Systems* **53**(1), 65–84 (2006)
8. Burkard, R., Dell’Amico, M., Martello, S.: *Assignment problems: revised reprint*. SIAM (2012)
9. Cappelen, A.W., Tungodden, B.: Fairness and the proportionality principle. *Social Choice and Welfare* **49**(3), 709–719 (2017)
10. Caragiannis, I., Kurokawa, D., Moulin, H., Procaccia, A.D., et al.: The unreasonable fairness of maximum nash welfare. *ACM-TEAC* **7**(3), 1–32 (2019)
11. Chevaleyre, Y., Dunne, P.E., Endriss, U.e.a.: Issues in multiagent resource allocation. *Informatika* **30**(1), 3–32 (2006)
12. Conitzer, V., Freeman, R., Shah, N., Vaughan, J.W.: Group fairness for the allocation of indivisible goods. In: *Proc. of the AAAI Conference on AI*. vol. 33(1), pp. 1853–1860 (2019)
13. Cook, K.S., Hegtvedt, K.A.: Distributive justice, equity, and equality. *Annual review of sociology* **9**(1), 217–241 (1983)
14. Corbett-Davies, S., Goel, S.: The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023* (2018)
15. Cousy, K., Lujak, M., Salvatore, A., Fernández, A., Giordani, S.: On balancing fairness and efficiency of task assignment in agent societies. In: *Highlights in Practical Applications of Agents, Multi-Agent Systems, and Complex Systems Simulation. The PAAMS Collection. Communications in Computer and Information Science*, vol. 1678, pp. 95–107. Springer International Publishing (2022)
16. Fleurbaey, M., Maniquet, F.: *A theory of fairness and social welfare*, vol. 48. Cambridge U. Press (2011)
17. Ghodsi, A., Zaharia, M., Hindman, B.e.a.: Dominant resource fairness: Fair allocation of multiple resource types. In: *8th USENIX Symposium NSDI 11* (2011)
18. Giordani, S., Lujak, M., Martinelli, F.: A distributed multi-agent production planning and scheduling framework for mobile robots. *Computers & Industrial Engineering* **64**(1), 19–30 (2013)
19. Guo, H., Li, W., Deng, B.: A survey on fair allocation of chores. *Mathematics* **11**(16), 3616 (2023)
20. Hellman, D.: Measuring algorithmic fairness. *VLR* **106**(4), 811–866 (2020)
21. Jagtenberg, C., Mason, A.: Fairness in the ambulance location problem: maximizing the bernoulli-nash social welfare. Available at SSRN 3536707 (2020)
22. Kranich, L.: Resource-envy-free and efficient allocations: A new solution for production economies with dedicated factors. *Journal of Math. Econ.* **89**, 1–7 (2020)
23. Kraus, S., Plotkin, T.: Algorithms of distributed task allocation for cooperative agents. *Theoretical Computer Science* **242**(1-2), 1–27 (2000)
24. Kuhn, H.W.: The hungarian method for the assignment problem. *Naval research logistics quarterly* **2**(1-2), 83–97 (1955)
25. Kurzban, R., Duckworth, A., Kable, J.W., Myers, J.: An opportunity cost model of subjective effort and task performance. *Behavioral and brain sciences* **36**(6), 661–679 (2013)
26. Lujak, M., Billhardt, H., Ossowski, S.: Distributed coordination of emergency medical service for angioplasty patients. *Annals of Mathematics and Artificial Intelligence* **78**, 73–100 (2016)

27. Lujak, M., Giordani, S., Omicini, A., Ossowski, S.: Decentralizing coordination in open vehicle fleets for scalable and dynamic task allocation. *Complexity* **2020**(1047369), Article ID 1047369 (2020)
28. Lujak, M., Sklar, E., Semet, F.: On multi-agent coordination of agri-robot fleets. In: CEUR Workshop Proc. of The 11th International Workshop on Agents in Traffic and Transportation (ATT 2020) held in conjunction with the 24th European Conference on Artificial Intelligence (ECAI 2020). vol. 2701(12) (2020)
29. Moulin, H.: Fair division and collective welfare. MIT press (2004)
30. Moulin, H.: Axioms of cooperative decision making. No. 15 in *Econometric Society Monographs*, Cambridge U. P. (1991)
31. Mulgan, T.: Understanding utilitarianism. Routledge (2014)
32. Nanda, V., Xu, P.e.a.: Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. In: Proc. of the AAAI Conference on Artificial Intelligence. vol. 34(2), pp. 2210–2217 (2020)
33. Nelson, L.M.: Collaborative problem solving. *Instructional design theories and models: A new paradigm of instructional theory* **2**(1999), 241–267 (1999)
34. Padala, M., Gujar, S.: FNNC: achieving fairness through neural networks. In: Proc. of the Twenty-Ninth Int. Joint Conf. on Artif. Int. IJCAI-20 (2020)
35. Ramezani, S., Endriss, U.: Nash social welfare in multiagent resource allocation. In: *Agent-mediated electronic commerce. Designing trading strategies and mechanisms for electronic markets*, pp. 117–131. Springer (2009)
36. Rohbogner, G., Hahnel, U.J., Benoit, P., Fey, S.: Multi-agent systems' asset for smart grid applications. *Computer Science and Information Systems* **10**(4), 1799–1822 (2013)
37. Sampat, A.M., Zavala, V.M.: Fairness measures for decision-making and conflict resolution. *Optimization and Engineering* **20**(4), 1249–1272 (2019)
38. Schappe, S.P.: Understanding employee job satisfaction: The importance of procedural and distributive justice. *J. of Business and Psyc.* **12**(4), 493–503 (1998)
39. Sen, A.: Collective choice and social welfare. In: *Collective Choice and Social Welfare*. Harvard University Press (2017)
40. Shrestha, Y.R., Yang, Y.: Fairness in algorithmic decision-making: Applications in multi-winner voting, machine learning, and recommender systems. *Algorithms* **12**(9), 199 (2019)
41. Suksompong, W.: Constraints in fair division. *ACM SIGecom Exchanges* **19**(2), 46–61 (2021)
42. Zhao, Y., Zheng, K., Guo, J., Yang, B., Pedersen, T.B., Jensen, C.S.: Fairness-aware task assignment in spatial crowdsourcing: Game-theoretic approaches. In: 2021 IEEE 37th ICDE. pp. 265–276. IEEE (2021)

Marin Lujak is a Distinguished Researcher Lecturer at the University Rey Juan Carlos. His research focuses on AI and distributed optimization and coordination of multi-agent and cyber-physical systems. In more than 70 peer-reviewed publications, he addresses societal challenges like smart transport, emergency management, and multi-robot coordination.

Alessio Salvatore is a Postdoctoral Researcher at Institute for Systems Analysis and Computer Science - Italian National Research Council. His main research interests are Mixed Integer Linear Programming and the development of heuristic algorithms with applications in transportation, logistics and scheduling.

Alberto Fernández, a Full Professor at the University Rey Juan Carlos, has over 20 years of research experience in the field of Artificial Intelligence. With more than 120 peer-

reviewed publications, his research activities are centered particularly in areas such as multi-agent systems and knowledge representation.

Stefano Giordani, a Full Professor of Operations Research at the University of Rome Tor Vergata, is renowned for his extensive research contributions in combinatorial optimization and its applications in scheduling, transportation, and logistics. With over 100 peer-reviewed publications, his work has significantly advanced the field.

Kendal Cousy is a PhD student in multi-agent coordination for agricultural robotics. He pursued research through a cotutelle program between the University Rey Juan Carlos and the University of Rome Tor Vergata.

Received: January 19, 2023; Accepted: October 10, 2023.

Comparing Reinforcement Learning Algorithms for a Trip Building Task: a Multi-objective Approach Using Non-Local Information*

Henrique U. Gobbi, Guilherme Dytz dos Santos, and Ana L. C. Bazzan

Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre, RS, Brazil
{hugobbi,gdsantos,bazzan}@inf.ufrgs.br

Abstract. Using reinforcement learning (RL) to support agents in making decisions that consider more than one objective poses challenges. We formulate the problem of multiple agents learning how to travel from A to B as a reinforcement learning task modeled as a stochastic game, in which we take into account: (i) more than one objective, (ii) non-stationarity, (iii) communication of local and non-local information among the various actors. We use and compare RL algorithms, both for the single objective (Q-learning), as well as for multiple objectives (Pareto Q-learning), with and without non-local communication. We evaluate these methods in a scenario in which hundreds of agents have to learn how to travel from their origins to their destinations, aiming at minimizing their travel times, as well as the carbon monoxide vehicles emit. Results show that the use of non-local communication reduces both travel time and emissions.

Keywords: reinforcement learning, multi-agent systems, multi-objective reinforcement learning, route choice.

1. Introduction

Recent publications in the area of multi-agent systems are showing the value of using multi-objective reinforcement learning (RL) in agents' decision making processes. The rationale here is that many tasks are better dealt with when multiple – possibly conflicting objectives – are considered. Although this poses more challenges to RL, especially when more than few agents interact, such formulation is useful in real-world problems as for instance making decision regarding trips in traffic networks. In this domain, multiple drivers must learn to reach their destinations, starting at their given origins. Depending on the formulation of the RL task, agents construct their routes by making decisions about which link to follow, once they find themselves at given locations. Thus, locations are states and actions are selection of links. This way, agents learn how to construct a route from their origins to their destinations.

Usually, for such learning task, a single objective is considered, namely minimizing travel times. However, frequently, there are other objectives to be considered. For instance, there are works that optimize two objectives – toll and travel time – such as [14, 21, 22]. These works are based on methods that are centralized and do not involve RL.

* Extended version of a paper presented at the ATT 2022 workshop.

Route choice using travel time and toll by means of RL is addressed by [8]; however this work deals with agents selecting among k pre-computed routes that take the agents from their origins to their destinations. This means that the RL task involves only one state (stateless RL), namely the origin node, where an agent makes a decision (select one of the k routes). Then the agent follows the selected route without making further decisions during the trip. For this kind of problem, the work described in [8] extends a Bandit algorithm like UCB [1], in order to account for multiple objectives and for multiple learning agents.

In contrast to the aforementioned works, in the present paper, each agent or driver not only performs its optimization process locally (in a decentralized way), by means of multi-objective RL (MORL), but also it builds its trip by making decisions at each intersection (that function as states). Hence, the underlying learning task is state-based.

Other characteristics of the problem we deal with is that it involves many agents competing for resources. This also poses challenges to RL methods, even if only one objective is considered. Specifically, the fact that there are many agents learning simultaneously causes the environment to be non-stationary.

As these features make the learning task hard, in this paper we aim at investigating the benefits of communication among the various actors (vehicles, elements of the road infrastructure) and, in particular, the role of using non-local information. The rationale for this research question relates to benefits reported in the area of new communication technologies such as vehicle to infrastructure (V2I) communication ([9, 10, 17–19]). We posit that, by effectively using communication-based approaches, congestion and, consequently, emissions can be reduced.

In short, a decentralized RL-based approach to routing involves the following issues: (i) agents learning simultaneously result in non-stationarity; (ii) there are potentially two or more objectives to be optimized; (iii) the underlying learning task is modeled as a stochastic game, where states (in which decisions about which link to take are made) are the intersections of the networks; (iv) communication among the various actors in the road network.

The remainder of this paper is organized as follows. The next section discusses background concepts and gives an overview on the related work. Section 3 details the proposed method. Its evaluation in a proof-of-concept scenario is discussed in Section 4. Concluding remarks and future directions are given in Section 5.

2. Background and Related Work

In this section, we briefly introduce underlying concepts on RL, as well as on traffic assignment, route choice, and routing (including multi-objective variants), as well as on V2I and other types of communication that are possible in a road network.

2.1. Reinforcement Learning

In RL, an agent learns how to act in an environment, by receiving a feedback (reward) that measures how its action has affected the environment. The agent does not a priori know how its actions affect the environment, hence it has to learn this by trial and error, which

characterizes an exploration phase. Such phase may be very noisy in multiagent scenarios, given that all agents are learning simultaneously and hence have to adapt to others' exploration processes. However, agents should not only explore; in order to maximize the rewards of their actions, they also have to exploit the gained knowledge. Thus, there must be an exploration-exploitation strategy that is to be followed by the agents. One of these strategies is ε -greedy, where an action is randomly chosen (exploration) with a probability ε , or, with probability $1-\varepsilon$, the best known action is chosen, i.e., the one with the highest value so far (exploitation).

In the exploitation phase, at each interaction, it is assumed that an agent has sensors to determine its current state and can then make an action. The reward perceived or received from the environment is used to update its policy, i.e., a mapping from states to actions. This policy can be generated or computed in several ways. For the sake of the present paper, we concentrate on a model-free, off-policy algorithm called Q-learning or QL [23] and its extensions. In QL, the value of a state s_t and action a_t at time t is updated based on Eq. 1, where $\alpha \in [0, 1]$ is the learning rate, $\gamma \in [0, 1]$ is the discount factor, s_{t+1} is the next state and r_t is the reward received when the agent moves from s_t to s_{t+1} after selecting action a_t in state s_t .

The reason for using QL is based on the fact that it has proven both effective and efficient, i.e., leads to agents learning routes that quickly optimize some quantity (normally, but not only, travel time). See references in the next section.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_a(Q(s_{t+1}, a)) - Q(s_t, a_t)) \quad (1)$$

RL can be employed to compute how drivers learn how to go from A to B in a transportation network. The next section thus introduces this learning task.

2.2. How to Travel from A to B?

In this section, we first discuss the terminology that is related to this problem. Then, we introduce concepts underlying the definition of an urban network (topology), as well as the demand side. Finally, we discuss the need for multi-objective approaches.

Given a road network and the demand for its use (number of trips per unit of time), the task of finding out how each unit of a demand realizes its travel needs can be solved in different ways, although most of them seek to compute the so-called user equilibrium, i.e., the condition in which no user is better off by changing its route. We recall that, in the literature, demand can be expressed as user, trip, traveller, agent, or vehicle, depending on the community.

In the transportation community this problem is normally known as the traffic assignment problem (TAP), which is solved in a centralized fashion, mostly via a macroscopic approach using a volume-delay function (VDF), which estimates the travel time as a function of the volume of trips; see Chapter 10 of [12] for details. In the computer science community, that task is frequently solved by means of RL in a decentralized way, in the sense that agents learn how to reach their destinations, by performing experiments (exploration) until they learn their respective routes (which is equivalent to the condition under the user equilibrium). This task admits two variants: in the first, an agent knows a set of precomputed routes (or is able to compute them), and its task is to choose a route among them (again by means of experimenting with several selections of actions

or routes). Once such choice is done, an agent travels the route, without changes during the trip. Examples of this approach are [11, 15] among others. The second variant departs from the assumption that agents are given a set of k shortest routes. Instead, agents make choices during the trip (e.g., at each intersection, or at each important point of decision), thus building the actual route while traveling, i.e., they decide which link to follow once they find themselves in each vertex. This is the approach followed in [2, 19] and also in the present paper.

In terms of RL, these two variants have a key difference. While the latter is a classical RL task with a set of states (e.g., each decision points), and a set of actions, the former belongs to the stateless RL front, where agents basically need to select an action at their respective origin node or state. Moreover, this learning task resembles Bandit algorithms such as UCB [1]. Henceforth, we refer to the former as route choice (as the agent simply chooses a route and remain on it), while the latter is referred here as routing to convey the idea that the trip is built during the actual routing task.

Regarding approaches that were already proposed to tackle this problem, in the literature, there are various approaches to solve the TAP; here we refer only to those that address more than one objective such as [5, 14, 21, 22]. Others can be found in [3].

Regarding route choice, the reader is referred to [11, 15]; in particular, multi-objective route choice is tackled by [8]. All these works use a macroscopic approach in the sense that VDFs are used to provide a reward for each agent based on the flow on a given route.

Routing is less common in the literature but some works appear in [6, 17–19]. While the former employs a VDF to compute rewards (thus, a macroscopic approach), the last two use a microscopic simulator, where vehicles realize the actual driving movements. As such, rewards (e.g., travel times) are given by the simulator itself and correspond to a more realistic reward (as opposed to the estimation provided by a VDF).

Next we briefly review the concept of transportation network and the demand that uses it. Formally, a traffic network can be modeled as a graph $G = (V, L)$, where V is the set of vertices or nodes that represent the intersections, and L , the set of links that describe the segments connecting a pair of vertices. In the network, trips are distributed among a set of origin-destination (OD) pairs; each corresponding to a certain demand for trips. These then translates into flows on the various links.

In all methods it is assumed that agents (drivers) are rational, thus each selects the route or link with the least perceived individual cost, in order to travel from its origin to its destination. Several factors may influence this decision, such as travel time, distance, monetary cost (e.g., toll), fuel consumption, battery, emission of pollutants, etc.

Traditionally, multi-objective traffic assignment is modeled by using a linear combination of the various objectives, as proposed, for instance, by Dial in [5] for a bi-objective assignment. Such a linear combination has some drawbacks. For instance, efficient routes may be missed, as discussed in [21]: in a nutshell, the key point is that algorithms that use a linear combination only identify supported solutions at extreme points, while there may be other efficient solutions that are not considered in that group but could be preferred by some users.

Hence, it is necessary to have alternative solutions. One issue that arises is that, in a multi-objective scenario, there is a set of efficient solutions rather than just one. Different solutions have been proposed in the transportation and optimization communities; see for instance [5, 14, 21, 22]. We remark that all these methods rely on a set of (pre-computed) k

shortest paths and are centralized. Thus, they are not useful in a multiagent environment where each agent should learn by its own experience using RL, nor they fit a state-based RL task, where agents learn at each intersection.

2.3. A Multi-objective Approach to Q-learning

In this section we discuss how to use QL when agents need to deal with more than one objective. For more details we refer readers to other sources, which also cover other approaches; see [7, 13].

In order to extend the aforementioned QL algorithm, [20] proposed Pareto Q-learning (PQL), which integrates the Pareto dominance relation into a MORL approach. PQL considers both the immediate reward vector and the set of expected future discounted reward vectors in order to compute the so-called Q-sets, which are composed of vectors. In PQL, the set of expected future discounted reward vectors relies on a function, called ND (from non-dominated), that finds those vectors that correspond to the possible future states, and which are not Pareto-dominated.

Eq. 2 shows how Q-sets are calculated. $\bar{R}(s, a)$ denotes the immediate reward vector and $ND_t(s, a)$ is the set of non-dominated vectors in the next state s , which is reached by performing action a at time step t . $\bar{R}(s, a)$ is added to each element of $\gamma ND_t(s, a)$. When a is selected at s , both terms are updated. $\bar{R}(s, a)$ is updated according to Eq. 3, where \vec{R} is the new reward vector and $N(s, a)$ is the number of times action a was selected in s . $ND_t(s, a)$ is updated as shown in Eq. 4, using the non-dominated vectors in the \tilde{Q}_{set} of every action a' in the next state s' .

$$\tilde{Q}_{set}(s, a) = \bar{R}(s, a) \oplus \gamma ND_t(s, a) \quad (2)$$

$$\bar{R}(s, a) = \bar{R}(s, a) + \frac{\vec{R} - \bar{R}(s, a)}{N(s, a)} \quad (3)$$

$$ND_t(s, a) = ND(\cup_{a'} \tilde{Q}_{set}(s', a')) \quad (4)$$

PQL learns the entire Pareto front, finding multiple Pareto optimal solutions, provided that each state-action pair is sufficiently sampled. This algorithm is not biased by the Pareto front shape (algorithms that find a single policy and use scalarization cannot sample the entire Pareto front if it is non-convex) or a weight vector (it guides the exploration to specific parts of the search space).

Note that PQL assumes that the environment is deterministic, hence it is not directly useful for environments in which the presence of multiple agents learning simultaneously causes non-stationarity, as for instance the route choice domain we discuss ahead. Therefore, we adapted PQL to deal with multiple objectives and with multiple agents (thus, a non-stationary environment). Henceforth this adapted algorithm is denoted as aPQL.

2.4. Communication in Road Networks

As aforementioned, communication in road networks (among vehicles, between vehicles and infrastructure, etc.) has started to receive attention also in the traffic engineering community. The reader is referred to [9] (focusing on how autonomous vehicles and connected

vehicles are expected to increase the throughput of highway facilities, as well as improve the stability of the traffic stream), and to [10] (for applications).

The use of this kind of technology was investigated by us previously in [17–19], where we have connected it to MARL, in order to investigate how V2I communication could help drivers in RL-based trip building or routing. In these works, the infrastructure is able to communicate with the vehicles, both collecting information about their most recent travel times (on given links), as well as providing them with information that was collected from other vehicles. However, besides considering a single objective only, links in the infrastructure only exchange information if they are connected by a junction, i.e., only local information is considered, whereas in the present paper we also consider that those links share non-local information that they acquire from links that are similar to them. For this, as detailed ahead, we employ a virtual graph where nodes are the (road) links and edges exist between any two links that have similar values in terms of travel time and emission. This kind of approach was also used by [4]. However, that work left the use of multiple objectives as an open direction.

3. Methodology

This section details our methods. We start by defining the virtual graph that connects elements of the road network, thus allowing them to communicate and exchange information that will then be passed to drivers, to help these to make decisions. These are the topics presented in the next two subsections. We then formalize the learning task itself, ending the section recapitulating the main points of the approach.

3.1. Terminology: Road Network and Virtual Graph

We start by stressing that we deal with two sorts of graphs. First, a road network is a (planar) graph $G = (I, L)$, where I is the set of intersections, and L is the set of links. In short, G represents the road network and mirrors its topology. For example, as discussed ahead in more details, in Fig. 2 G is a 5×5 grid, in which intersections A4 and B4 are connected by a link $A4 \leftrightarrow B4$.

The other graph refers to non-local connections among two (not necessarily physically close) links $l_1 \in L$ and $l_2 \in L$ (as, e.g., $A4 \leftrightarrow B4$ and $C1 \leftrightarrow D1$ in Fig. 2). A connection in this graph arises if both show similar patterns. We call this a virtual graph denoted by $VG = (L, E)$, where L is the set of links (note that now links act as vertices in VG), and E is the set of edges that connect two links that have similar patterns, as described next in more details.

In order to apply non-local communication to share information across the network, VG connects links in the network that share similar attributes. In VG , every node is a link $l \in L$ at a specific time step. An edge is only created when two of these nodes share similar attributes. This happens as follows: first of all, data related to some attributes of the links is gathered along time. This data can either be historical data or be collected in a simulation. This way, various attributes, such as travel time, link occupancy and CO (carbon monoxide) emission are collected at each time step. The data is then normalized and used to compare every link at a time step against every other link, checking if the absolute difference between the values of their attributes is within a defined threshold Δ .

If two links have attributes whose values are within Δ , an edge between them is added to VG .

Fig. 1a shows an instance¹ of such a virtual graph (corresponding to the network G depicted in Fig. 2), whereas, for a better visualization, Fig. 1b shows a zoom of that graph, where some relationships among similar links can be better seen. The labels of the vertices are formed by the link ID plus the time interval in which their values were found to be similar.

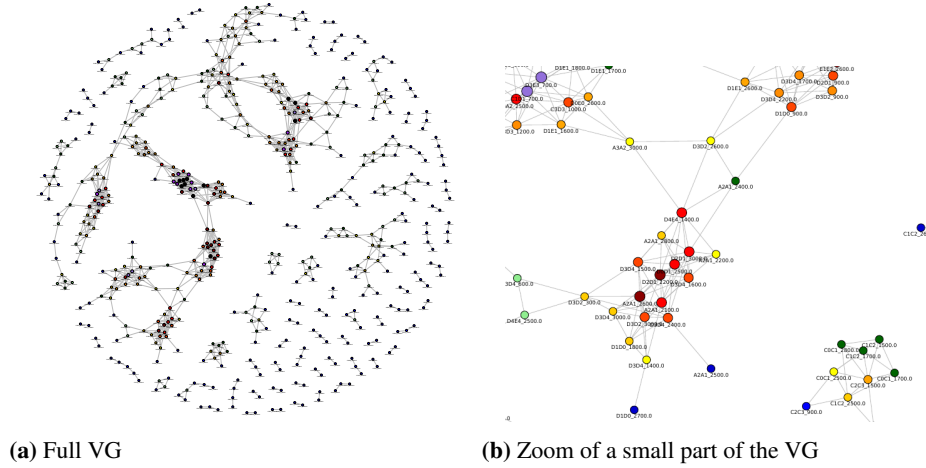


Fig. 1. Instance of a virtual graph VG

3.2. Communication Using the Virtual Graph

The central idea of the communication using the virtual graph is to extend the communication between driver agents and elements of the road infrastructure (mostly the links via communication devices installed on them), so that such elements send information about the status of the links to drivers. For example, assume that l_1 and l_2 are neighbors in the VG because, despite not being close, they share similar values for some selected attributes. This means that they both exchange information, which is then passed to those driver agents that intend to use such links.

More specifically, an intersection collects information from its incoming links, defined in the physical road network G . Additionally, given that these incoming links may have virtual neighbors in the virtual graph VG , information about their virtual links neighbors are also passed to the links. Once an intersection i has collected such information, it updates a table in which the last 30 entries are kept in a FIFO way, for each attribute. This window size was used in [18] for the same scenario. In the present paper, an intersection stores information also about travel time and CO emission.

¹ For sake of clarity, we have generated such figure using an arbitrary low number of time steps; the actual graph is denser than represented in that figure.

The intersection then communicates to each nearby driver agent an aggregation of those values kept in the tables, i.e., potential rewards that the driver may obtain if selecting each action in that particular state. The agent then perceives this information as expected rewards for the actions available to it.

To summarize, the purpose of the virtual graph VG is to indicate which links in the road network should exchange information about the network during communication at various moments throughout the simulation. The VG relates links that share similar patterns, therefore these links will exchange information with each other during the simulation in an effort to accelerate and improve the learning of the agents, given that the experiences of the agents are widely shared across the network.

3.3. The Learning Task Formalized as a Markov Decision Process

In this section, we discuss how the learning task was formalized as an MDP (see Section 2.1).

Recall that driver agents build their routes by making decisions about which link to follow, when finding themselves in a given intersection. Thus, in this routing learning task, each intersection $i \in I$ within a traffic network G is a state s , and the actions correspond to the links an agent might take when in s (i.e., links that leave the intersection).

Regarding the environment, the transition model is implemented by a microscopic traffic simulator, which moves vehicles along the network.

As for the rewards, since we deal with a multi-objective scenario, the rewards returned by the simulator are assembled in a vector; this way, the reward function definition is $R : S \times A \rightarrow \mathbb{R}^n$, where n is the number of objectives. As presented ahead in Section 4.2, in the present paper we deal with agents aiming at optimizing two objectives: travel time and carbon monoxide emission. However, the proposed method is general and can be used if more objectives are taken into account.

At this stage, a note about action selection is necessary. Originally, to deal with reward being a vector, in the PQL algorithm (i.e., as proposed in [20]), every pair state-action (s, a) is associated with a set of non-dominated points that represents the Pareto front. Q-sets are computed as in Eq. 2 and are used to compute a hypervolume of the Pareto front. This is then used to determine the best action. In our case, this was not effective given that the Q-sets tend to contain few points only, thus computations based on hypervolume tend to be ineffective. Therefore, we had to modify the action selection strategy. Instead of using the hypervolume of the Pareto front to determine the best action, we let each agent select randomly (with uniform probability, at each decision point, and independently of one another) which objective is to be optimized at that given step. This seems to be a fair substitution for the hypervolume and avoids biasing towards one objective. Once a particular objective is randomly drawn, then an agent uses the ε -greedy strategy to select an action, given the Q-set. Note that the approach continues to be multi-objective, as all objectives have their values updated after a given action was selected. To differentiate from PQL, we refer to our approach as aPQL.

3.4. The Whole Picture

Algorithm 1 shows how our method works for each agent, in terms of how it deals with learning when getting additional information gain from communication. Each agent trav-

Algorithm 1: Main Procedure (for each agent)

```

Data:  $M$ , origin, destination, learningParameters
1  $step \leftarrow 0$ 
2 agent starts trip in its origin
3 while  $step < M$  do
4   if agent has reached a decision point then
5      $\vec{r} \leftarrow$  reward from last link
6     communication agent and infrastructure
7     computation of potential rewards
8     update of knowledge base
9     if reached its destination then
10      agent restarts trip in its origin
11    else
12      agent chooses a new action (link to travel)
13    end
14  end
15   $step \leftarrow step + 1$ 
16 end

```

els the links and make decisions at the intersections, until $step = M$ (maximum number of steps).

Lines 4 through 14 correspond to the behaviour an agent has when it reaches an intersection. It perceives its reward from traveling through a link (Line 5): this reward is a vector composed by one element per objective (e.g., two objectives for travel time and CO emission). The agent then communicates with the intersection's infrastructure element (IE), informing its perceived reward and retrieving the expected rewards, as discussed in Section 3.2.

It then uses its own perceived reward, as well as the expected rewards communicated by the IE (Line 8) in order to update its knowledge base. For our experiments (Section 4), we compare QL with aPQL. An agent learning using QL thus updates its Q-Table (Eq. 1), while an agent using aPQL updates its ND set (Eq. 4).

If an agent reaches its destination, it gets reinserted at its corresponding origin to start a new trip. Otherwise, it chooses which link to travel next. This action selection uses the ε -greedy strategy, as discussed in Section 3.3. This means the agent will take the action which maximizes its future gains with a probability of $(1 - \varepsilon)$, or take a random action with a probability of ε .

4. Experiments and Results

4.1. Network and Demand

To test the approach, we used a 5×5 grid network depicted in Fig. 2. The demand corresponding to the OD (origin-destination) matrix is shown in in Table 1. Each link shown in Figure 2 is two-way. We recall that both the network and this demand were used in [16]. Simulations were carried out using a microscopic traffic simulator, namely SUMO.

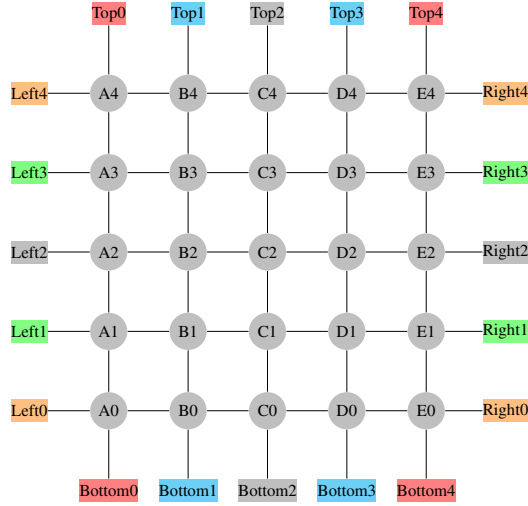


Fig. 2. 5x5 Grid Network (as in [19])

4.2. Specifying the MDP For This Scenario

In a traditional RL environment, at each time step, each agent finds itself in state s , chooses action a , receives a reward and transitions to a state s' . In the learning task at hand, time steps are controlled by the microscopic simulator and correspond to a unit of time (such as one second). This has important consequences. First, not all time steps count as decision-making steps; these only happen when an agent is at an intersection. The rest of the time steps are used (e.g., see plots ahead) just as clock units. Second, different agents find themselves either in decision-making states, or are moving along a link (thus, not updating their value functions), or have reached their destinations. In the latter case, an episode is finished *for that particular agent*. Since travel times and destinations are different for most of the agents, episodes are *not synchronous*. Therefore, in general, we do not refer to episodes; rather, we use the clock units or time steps of the simulator to depict time. To account for the simulation time, we use the parameter M (maximum number of steps), as introduced in Section 3.

Next, we detailed how we deal with multiple objectives in the reward function for this specific scenario.

In the exploration phase, due to uncoordinated action selection by the agents, congestion may occur in the road network, which impacts the reward of the agents, while also inserting noise in the learning process, which then leads to slow convergence. In order to speed up this process, and avoid agents wandering around the road network performing experimentation, two hyper-parameters were used. See [16] for details.

4.3. Parameters and Their Values

The parameters discussed in Section 3 were set as follows: maximum simulation steps $M = 40,000$ and threshold $\Delta = 0.005$. Two attributes were considered, namely travel

Table 1. Demand (nb. of trips) per OD-pair

OD-Pairs	Demand
Bottom0—Top4	51
Top4—Bottom0	51
Bottom4—Top0	51
Top0—Bottom4	51
Bottom1—Top3	43
Bottom3—Top1	43
Top3—Bottom1	43
Top1—Bottom3	43
Left1—Right3	43
Right3—Left1	43
Left3—Right1	43
Right1—Left3	43
Left0—Right4	51
Right4—Left0	51
Left4—Right0	51
Right0—Left4	51

time and CO emission. We remark that these values are provided by the traffic simulator (SUMO).

The method proposed was compared against: no learning (which means that the driver agents follow routes given by SUMO’s trip assignment); QL without the use of the virtual graph VG; QL combined with VG; aPQL without VG; and aPQL combined with VG. Recall that, when QL was used, the reward was a scalar, i.e., either travel time or CO emission (but not both together as it is the case with aPQL). Hence, we have run two distinct simulations: one in which the reward is based on the travel time alone, and another in which the reward is given by the CO emission. Following values reported in [16], for QL we have set the parameters – for both QL and aPQL – as follows: learning rate $\alpha = 0.5$, discount factor $\gamma = 0.9$, and $\varepsilon = 0.05$.

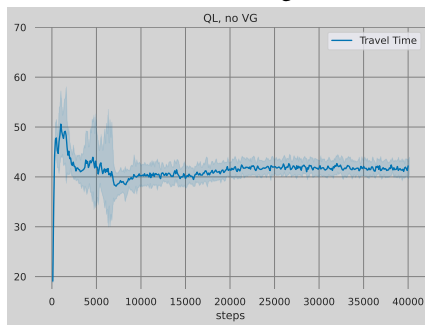
Its is also worth mentioning that, due to the stochastic nature of all methods we used for comparison, each of them were repeated 10 times. In the plots ahead, we show the mean values (and their standard deviations, as shadows).

4.4. Discussion of the Results

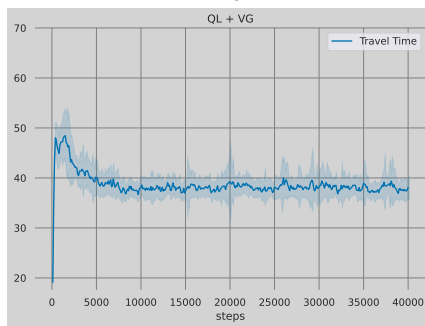
We first discuss plots that relate to travel time. For the sake of showing some baselines, in a first group (Fig. 3), we compare plots regarding QL with and without the use of the VG, and these against the case without learning. The latter is shown in Fig. 3a, where we can observe an average travel time per link around 50 time steps (or seconds). Note also that this plot is associated with a high deviation over the 10 repetitions.



(a) No learning



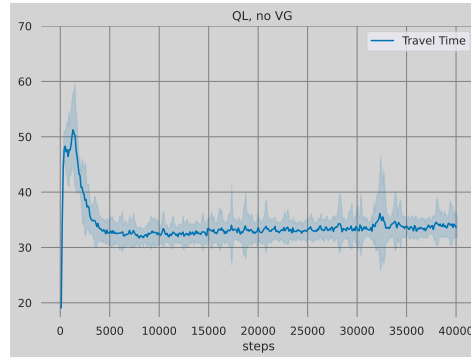
(b) QL



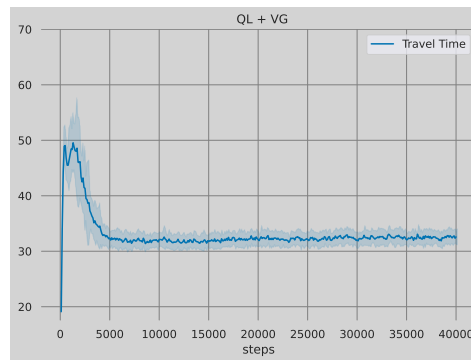
(c) QL plus virtual graph

Fig. 3. Travel time (average per link)

The cases in which QL was used appear in Fig. 3b and Fig. 3c. We can see that in the former, there is an improvement in performance (compared to the case without learning), as the travel time decreases to a value slightly over 40 when QL is used without the VG. When the VG is used, we see a further improvement. We stress that in both these cases, only travel time is being optimized so the use of the virtual graph is of reduced value.



(a) aPQL



(b) aPQL plus virtual graph

Fig. 4. Travel time (continued)

What interests us is indeed the case in which both objectives are considered together, i.e., the multi-objective approach that is proposed here. This case is shown in two plots in Fig. 4. Fig. 4a shows results for aPQL without the use of the VG, while Fig. 4b depicts the case in which VG is used. In both cases we see an improvement when compared with plots in Fig. 3: now the travel time converges to a value slightly over 30, with the latter – in which the VG was used – being better, as the travel time is lower and there are less oscillations.

It may not be completely obvious why in the multi-objective case we also observe a reduction in the travel time. We have conducted a (not yet extensive) investigation and concluded that the reason for such behavior is that, by minimizing emissions together with travel time, certain routes are chosen that distribute the vehicles better in the network

(as this reduces emissions). This in turn leads to lesser travel times. It remains to be investigated how correlated these two objectives are.

Similarly, Fig. 5 shows the results that relate to emission of CO. The plot in Fig. 5a shows the baseline when no learning algorithm was used. One observes a convergence to an emission of around 600 in mg/s. This value is then decreased when QL is used: to around 500 (Fig. 5b). When QL is combined with the VG (Fig. 5c), there is a further improvement.

Then, in Fig. 6, we show the plots for the multi-objective approach, where we can observe an even higher improvement, with the emission decreasing to values over 400. Finally, when the VG is combined with aPQL, the values decrease to below 400.

5. Conclusion and Future Work

Using RL to support agents making decisions that consider more than one objective poses challenges. We formulate the problem of multiple agents learning to travel from A to B in a traffic network as an RL task in which: (i) agents learning simultaneously result in non-stationarity; (ii) there are potentially two or more objectives to be optimized; (iii) the underlying learning task is modeled as a stochastic game, where states are intermediary locations in the networks (such as intersections), in which decisions are made about which link to take; (iv) non-local information is shared among elements of the infra-structure, which is then passed to driver agents.

To solve this task, we propose an adaptation in an existing algorithm, PQL [20]. This algorithm deals with more than one objective, but it was originally formulated for single-agent learning tasks, in which the environment is deterministic. In route choice, neither of these apply. Thus our adaptation addresses them. Moreover, we have combined this algorithm with a virtual graph that allows non-local communication.

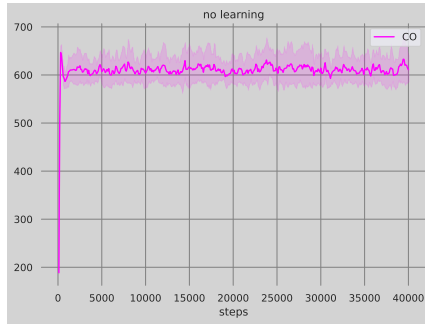
We have applied the proposed method to a scenario in which driver agents compete for a scarce resource (link usage), and have to learn how to travel from their origins to their destinations, aiming at minimizing their travel times, as well as the CO vehicles emit. Results show that the adapted algorithm, combined with the virtual graph is able to find routes that result both on lower travel times as well as emissions of CO.

As aforementioned, we intend to investigate the effect of a possible correlation between the two objectives we are using in the learning task.

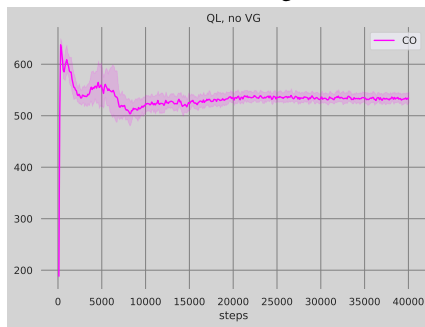
Acknowledgments. Ana Bazzan is partially supported by CNPq (grant 304932/2021-3). This work was partially funded by FAPESP and MCTI/CGI (grants number 2020/05165-1) and by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil, Finance Code 001), and also partially sponsored by the German Federal Ministry of Education and Research (BMBF), Käte Hamburger Kolleg Cultures des Forschens/ Cultures of Research.

References

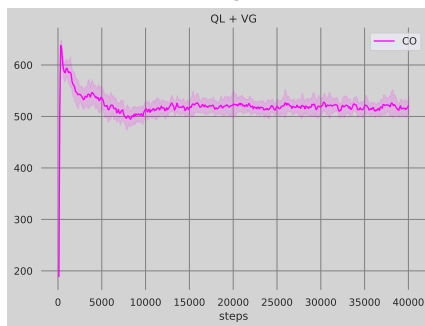
1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47(2/3), 235–256 (2002)
2. Bazzan, A.L.C., Grunitzki, R.: A multiagent reinforcement learning approach to en-route trip building. In: 2016 International Joint Conference on Neural Networks (IJCNN). pp. 5288–5295 (July 2016)



(a) No learning

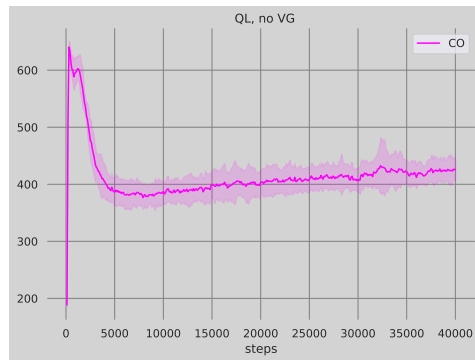


(b) QL

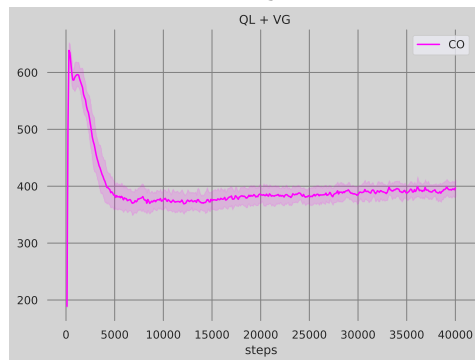


(c) QL plus virtual graph

Fig. 5. CO emission in mg/s (average per link)



(a) aPQL.



(b) aPQL plus virtual graph.

Fig. 6. CO emission in mg/s (continued)

3. Bazzan, A.L.C., Klügl, F.: Introduction to Intelligent Systems in Traffic and Transportation, Synthesis Lectures on Artificial Intelligence and Machine Learning, vol. 7. Morgan and Claypool (2013)
4. Bazzan, A.L., Gobbi, H.U., dos Santos, G.D.: More knowledge, more efficiency: Using non-local information on multiple traffic attributes. In: Proceedings of the KDMiLe 2022. SBC, Campinas (November 2022)
5. Dial, R.B.: A model and algorithm for multicriteria route-mode choice. *Transportation Research Part B: Methodological* 13(4), 311–316 (1979)
6. Grunitzki, R., Bazzan, A.L.C.: Combining car-to-infrastructure communication and multi-agent reinforcement learning in route choice. In: Bazzan, A.L.C., Klügl, F., Ossowski, S., Vizzari, G. (eds.) Proceedings of the Ninth Workshop on Agents in Traffic and Transportation (ATT-2016). CEUR Workshop Proceedings, vol. 1678. CEUR-WS.org, New York (July 2016), <http://ceur-ws.org/Vol-1678/paper12.pdf>
7. Hayes, C.F., Radulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., Howley, E., Irissappane, A.A., Mannion, P., Nowé, A., de Oliveira Ramos, G., Restelli, M., Vamplew, P., Roijers, D.M.: A practical guide to multi-objective reinforcement learning and planning. CoRR abs/2103.09568 (2021), <https://arxiv.org/abs/2103.09568>
8. Huanca-Anquise, C.A.: Multi-objective reinforcement learning methods for action selection: dealing with multiple objectives and non-stationarity. Master's thesis, Instituto de Informática, UFRGS, Porto Alegre, Brazil (2021), <http://hdl.handle.net/10183/231836>
9. Mahmassani, H.S.: Autonomous vehicles and connected vehicle systems: Flow and operations considerations. *Transp. Sci.* 50(4), 1140–1162 (2016)
10. Maimaris, A., Papageorgiou, G.: A review of intelligent transportation systems from a communications technology perspective. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 54–59 (2016)
11. de Oliveira, T.B.F., Bazzan, A.L.C., da Silva, B.C., Grunitzki, R.: Comparing multi-armed bandit algorithms and Q-learning for multiagent action selection: a case study in route choice. In: 2018 International Joint Conference on Neural Networks, IJCNN. pp. 1–8. IEEE, Rio de Janeiro (2018), doi.org/10.1109/IJCNN.2018.8489655
12. Ortúzar, J.d.D., Willumsen, L.G.: Modelling transport. John Wiley & Sons, Chichester, UK, 4 edn. (2011)
13. Rădulescu, R., Mannion, P., Roijers, D., Nowé, A.: Multi-objective multi-agent decision making: a utility-based analysis and survey. *Autonomous Agents and Multi-Agent Systems* 34 (04 2020)
14. Raith, A., Wang, J.Y., Ehrgott, M., Mitchell, S.A.: Solving multi-objective traffic assignment. *Annals of Operations Research* 222(1), 483–516 (2014)
15. Ramos, G.de.O., da Silva, B.C., Bazzan, A.L.C.: Learning to minimise regret in route choice. In: Das, S., Durfee, E., Larson, K., Winikoff, M. (eds.) Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017). pp. 846–855. IFAAMAS, São Paulo (May 2017), <http://ifaamas.org/Proceedings/aamas2017/pdfs/p846.pdf>
16. Santos, G.D.dos., Bazzan, A.L.C.: A multiobjective reinforcement learning approach to trip building. In: Bazzan, A.L., Dusparic, I., Lujak, M., Vizzari, G. (eds.) Proc. of the 12th International Workshop on Agents in Traffic and Transportation (ATT 2022). vol. 3173, pp. 160–174. CEUR-WS.org (2022), <http://ceur-ws.org/Vol-3173/11.pdf>
17. Santos, G.D.dos., Bazzan, A.L.C., Baumgardt, A.P.: Using car to infrastructure communication to accelerate learning in route choice. *Journal of Information and Data Management* 12(2) (2021), sol.sbc.org.br/journals/index.php/jidm/article/view/1935
18. Santos, G.D.dos., Bazzan, A.L.C.: Accelerating learning of route choices with C2I: A preliminary investigation. In: Proc. of the VIII Symposium on Knowledge Discovery, Mining and Learning. pp. 41–48. SBC (2020)

19. Santos, G.D.dos., Bazzan, A.L.C.: Sharing diverse information gets driver agents to learn faster: an application in en route trip building. PeerJ Computer Science 7, e428 (March 2021), peerj.com/articles/cs-428/
20. Van Moffaert, K., Nowé, A.: Multi-objective reinforcement learning using sets of Pareto dominating policies. J. Mach. Learn. Res. 15(1), 3483–3512 (Jan 2014)
21. Wang, J.Y.T., Raith, A., Ehrgott, M.: Tolling analysis with bi-objective traffic assignment. In: Ehrgott, M., Naujoks, B., Stewart, T.J., Wallenius, J. (eds.) Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems. pp. 117–129 (2010)
22. Wang, J.Y., Ehrgott, M.: Modelling route choice behaviour in a tolled road network with a time surplus maximisation bi-objective user equilibrium model. Transportation Research Part B: Methodological 57, 342–360 (2013)
23. Watkins, C.: Learning from Delayed Rewards. Ph.D. thesis, University of Cambridge (1989)

Henrique Uhlmann Gobbi is a B.Sc. graduate student in computer science at the Federal University of Rio Grande do Sul, Porto Alegre, Brazil, with an expected graduation date in mid-2025. His research interests primarily revolve around applying artificial intelligence to address real-world problems, in addition to investigating natural science phenomena. He has conducted research in the field of reinforcement learning applied to urban traffic mobility and is currently focusing on the use of artificial neural networks to model problems in neuroscience.

Guilherme Dytz dos Santos is a B.Sc. graduate student in computer science from Federal University of Rio Grande do Sul, Porto Alegre, Brazil, with expected graduation in early 2024. His academic pursuits are centered around research in reinforcement learning, machine learning, and artificial intelligence, all of which are applied to real-world problems in the realm of urban traffic optimization. His research focuses on harnessing the power of reinforcement learning to alleviate the challenges associated with high traffic demands in urban networks.

Ana Bazzan is a professor of Computer Science at the Institute of Informatics at the Universidade Federal do Rio Grande do Sul (UFRGS), where she leads the Artificial Intelligence Group. Her research focuses on multiagent systems, in particular on agent-based modeling and simulation, and multiagent learning. Since 1996, she has collaborated with various researchers in the application of multiagent systems. In recent years, she has contributed to different topics regarding smart cities, focusing on transportation. In 2014, she was General Co-chair of AAMAS (the premier conference in the area of autonomous agents and multiagent systems).

Received: December 10, 2022; Accepted: September 25, 2023.

Sustainability-Oriented Route Generation for Ridesharing Services [★]

Mengya Liu¹, Vahid Yazdanpanah², Sebastian Stein², and Enrico Gerding²

¹ Smart Data Group, AI Lab, Lenovo Research at Lenovo
No 10 Courtyard, Xibeiwang, East Road, Haidian District, Beijing, China, 100094
myliu26@lenovo.com

² Agents, Interaction and Complexity Research Group, University of Southampton
SO17 1BJ, Southampton, UK
v.yazdanpanah@soton.ac.uk
ss2@ecs.soton.ac.uk
eg@ecs.soton.ac.uk

Abstract. Sustainability is the ability to maintain and preserve natural and man-made systems for the benefit of current and future generations. The three pillars of sustainability are social, economic, and environmental. These pillars are interdependent and interconnected, meaning that progress in one area can have positive or negative impacts on the others. This calls for smart methods to balance such benefits and find solutions that are optimal with respect to all the three pillars of sustainability. By using AI methods, in particular, genetic algorithms for multiobjective optimisation, we can better understand and manage complex systems in order to achieve sustainability. In the context of sustainability-oriented ridesharing, genetic algorithms can be used to optimise route finding in order to lower the cost of transportation and reduce emissions. This work contributes to this domain by using AI, specifically genetic algorithms for multiobjective optimisation, to improve the efficiency and sustainability of transportation systems. By using this approach, we can make progress towards achieving the goals of the three pillars of sustainability.

Keywords: Ridesharing, Multiobjective Algorithm, Mobility-on-demand, Sustainable Transportation, Evolutionary Computation.

1. Introduction

Mobility-On-Demand (MOD) services traditionally aim to reduce the economic and environmental cost of transportation [11,1]. Roughly speaking, MOD promises to utilise the mobility capacity in a more efficient way (leading to reductions in the collective and individual economic costs), while also reducing emissions caused (e.g., by avoiding single-driver journeys). Therefore, MOD systems and methods to support their implementation directly contribute to Sustainable Development Goal (SDG) 11 and 13 on *sustainable cities and communities* and *climate action*, respectively. If MOD is widely adopted, cities will enjoy its environmental and economic benefits and take a step towards mitigating climate change. However, if such services merely focus on what is optimal from the service providers' perspective and ignore users' requirements, it will be difficult to encourage

[★] The initial idea behind this work was presented at ATT'22: Workshop Agents in Traffic and Transportation, July 25, 2022, Vienna, Austria [15].

users to move towards this service and ignore the comfort of using personal vehicles. Such a sustainability-oriented transition necessitates looking not only at operators' (economic) criteria, but also evaluating how a particular routing choice may affect individuals, e.g., via their total travel or waiting times. In this work, we argue that *sustainable ridesharing* needs to capture all the three pillars of economic, environmental, and social sustainability [19] and aim for routing solutions that are balanced with respect to all three aspects. The economic and environmental aspects call for minimising respective costs, both at a collective level, but also (to ensure fairness) for all the riders. Finally, the social aspect requires considering fairness in distributing tasks among drivers such that riders receive a balanced workload. Thus, addressing the routing problem in sustainable ridesharing requires a multiobjective approach that captures potential trade-offs among different aspects of sustainability for generating sustainable routing options.³

As discussed in related work, e.g., [21,1,25], the multidimensionality and complexity of the ridesharing routing problem, and, in our case, in view of the three pillars of sustainability, result in inapplicability of exact multiobjective optimisation techniques and justifies using genetic algorithms (GA). While [12] explored adopting reinforcement learning for multiobjective optimisation, this work considers GA to provide a diverse range of solutions (for a diverse set of users). Multiobjective GA allows capturing various objectives with fewer compromises regarding scalability [10]. In particular, we use a form of the Non-dominated Sorting Genetic Algorithm (NSGA) that is proven to be effective in various mobility settings [8].

Against this background, for the first time in this work, we capture the *social* and *environmental* aspects of sustainable routing in ridesharing and use genetic algorithms for generating routing options that consider all three pillars of sustainability. This is the first approach that integrates these two pillars of sustainability into traditional models of (purely) *economic* sustainability and generates routing solutions under six sustainable ridesharing objectives: travelling time, waiting time, overall/excess distance, travel cost, total emission, and working time balance. The list of sustainable routing options can be used in a “*user participation*” phase (e.g., in ridesharing services), where riders can select their desired route from a list of options. Using our approach, service providers can generate routing options that balance all the three pillars of economic, environmental, and social sustainability. In addition, they can provide routing options that reflect riders' preferences (e.g., by focusing on the environmental dimension). This is a step towards integrating equitability and user participation [2] into sustainable ridesharing practices.

In the remaining sections of this paper, we present a detailed explanation of the proposed ridesharing algorithm in Section, following which, in Section 3, is a brief introduction of the genetic algorithm (NAGA3) that we adopted; Section 4 explains the conducted experiments and illustrates the experiments results with respect to multiple aspects, and at the end, Section 5 concludes the contribution of this work.

³ In view of human-centred AI techniques and the need for developing trustworthy human-AI partnerships [20], we see sustainable ridesharing as an inherently sociotechnical problem and argue that its acceptance by society depends on the ability to capture all the three aspects of economic, environmental, and social sustainability.

2. Sustainable Ridesharing

The ridesharing routing problem is to allocate a given set of riders to vehicles with respect to different objectives. Each rider requires a ride from its starting point to its destination, along with a specified earliest departure time for taking a ride. Each vehicle can take a limited number of riders aboard at the same time, excluding the driver, which we refer to as its capacity. And the driving costs and capacity of a vehicle are associated with its type. A solution to the ridesharing routing problem is an arrangement that sends vehicles to pick up riders at their starting point, and then drops them off at their destination. The objectives evaluate the efficiency of a solution. In the following, we present the mathematical notations used for modelling the ridesharing problem and developing our approach.

2.1. Ride Requests

In the ridesharing routing problem, all the riders post their ride requests at the beginning. Let R be the set of posted requests and r represent a single request. To locate riders, we adopt a graph structure to model the real-world map, i.e., a map graph is $G = (N, E)$, where nodes in N represent the intersections on a real-world map and edges in E that link nodes together represent the roads between intersections. In general, to represent an intersection, a node is in the form of a tuple marking the latitude and longitude of the intersection. Thereby, let $r(s, t, u)$ denote a rider's request for a ride from node s to node t along with an earliest time for the rider to leave, u .

2.2. Features of Vehicles

To model the sustainability of the ridesharing routing problem from economic, environment and social aspects, this work considers 3 features of a vehicle as well as its location. Let V be the set of available vehicles and $v(s, t, p, e, c)$ denote a vehicle that starts working at node s , returns to node t at the end of its service with a physical capacity of p , emission level of e and a travelling cost per kilometre, c , including the driver wage, vehicle maintenance and fuel costs. Regardless of the difference of vehicle brands, there is a positive correlation between c and p . Hence, we assume that $c = \alpha \times p$ for a vehicle and α varies according to the type of vehicle. For a pessimistic estimation, we use $\alpha = 1$ in our experiments. Besides, the emission level of a vehicle e is a vector, and the i th element in the vector, e_i , denotes the emission rate of a vehicle when there are i riders aboard, since different numbers of riders aboard cause different emission rate. Specifically, in this work, the emission of a vehicle generally includes greenhouse gas (GHG) and air pollution. With respect to the reports from the UK's Department for Transport [16] and National Atmospheric Emission Inventory (NAEI) [23], and the EU standard vehicle emissions calculator, COPERT [9], the GHG and air pollution emission of a vehicle per kilometre are mainly related to the fuel and type of a vehicle, but the emission of GHG also depends on the number of passengers. Therefore, we model the emission level of a vehicle as a vector, where each element represents a emission rate associated with a number of riders on board. Notice that we also assume that all vehicles will drive at the same speed to simplify the problem. This is because the experimental data used in this work are city driving records which is more believed to be limited by the traffic condition instead

Table 1. Estimated Features for Different Types of Vehicle

	Vehicle Type	Fuel	Capacity	Vector of Emission Level	Cost
1	Mini Car	Petrol	1	$\langle 1, 2 \times 0.9 \rangle$	1
2	Car	Electric	3	$\langle 1/3.4, 2/3.4 \times 90\%, 3/3.4 \times 90\%^2, 4/3.4 \times 90\%^3 \rangle$	3
3	Medium Car	Petrol	6	$\langle 1, 2 \times 90\%, \dots, 7 \times 90\%^7 \rangle$	10
4	Large Car	Petrol	10	$\langle 1, 2 \times 90\%, \dots, 11 \times 90\%^{10} \rangle$	10

of the type of vehicles. This setting can be simply extended to simulate a dynamic speed by varying the speed in a range of minimum to maximum urban/legal speed.

The features of a vehicle, such as capacity, emission level and travelling cost, depend on its type. We consider 3 types of general passenger vehicles according to the vehicle categories specified on the UK Driving Licence Categories [22]: Small cars, medium-sized vehicles, large vehicles⁴. Table 1 lists their features. To simulate the emission level of different types of vehicles, we use the *car* type with a petrol engine and one passenger on board as the standard, and assume it emits 1 unit of greenhouse gas and air pollution per kilometre (e.g., 1 unit could be 100g of CO₂) and costs 1 price unit per kilometer (e.g., \$1). According to the Transport and Environment Statistics [16], “an average petrol car emits around 4 times more per passenger than the equivalent journey by coach, or 3.4 times more per passenger emitted by the average electric car”, and “maximising the number of people per vehicle can reduce emissions per person”. Hence, we assume that the average emission per passenger reduces by 10% when the number of passengers aboard increases and the cost of vehicles are related with its capacity. The emission column in Table 1 lists the emission vector for each vehicle type where elements in the vector are emissions of a vehicle in the order of 0 passengers to its full capacity.

Our focus in this work is to demonstrate the impact of emissions of vehicles, and we are aware of the existence of other types of vehicles, such as motorcycles, and different types of emission calculators [24]. The types of vehicles and the emission estimation considered in this work are standard types and presented for the purpose of showing the performance of the approach.

2.3. Computational Complexity

To understand the complexity of ridesharing problem, we can analyse the number of arrangements and driving routes of all vehicles. First, the number of possible arrangements is $|V|^{|R|}$ since individual riders have $|V|$ options for their rides without considering feasibility, i.e. constrains such as the capacities of vehicles and the relative order of starting points and destinations to stop by. Regrading an arrangement, for each vehicle arranged to offer rider to τ passengers, there are $2\tau!$ different driving routes to stop by all the starting points and destinations of those riders. To simplify the calculation, assume the computational complexity of justifying whether a route is feasible and calculating objectives are $O(|V|)$. In addition, given multiple objectives, assume the best weights of multiple objectives for a ridesharing problem is preserved. Thereby, the computational complexity of

⁴ we exclude minibuses and buses [6]. Since they have stable routes and we do not consider asking riders to change vehicles during their rider, they left no space for picking riders at their starting points.

ridesharing problem to find an optimal arrangement regarding the objectives is

$$O(R, V, Objectives) = O(|V| + |V|^{|R|} \times \prod_{\tau_1, \dots, \tau_{|V|}} 2\tau_i!),$$

where $\sum_{i=1}^{|V|} \tau_i = |R|$.

It will be expensive to find one optimal arrangement, especially when no prior knowledge is given about the method to balance multiobjectives. In the following sections, we will explain our design to filter out infeasible arrangements and adopt an genetic algorithm to efficiently generate diverse arrangements without requiring prior knowledge about multiobjectives.

2.4. Solution Design

Our intelligent routing approach is based on genetic algorithms [17]. First, we model an arrangement that a vehicle picks up and drops off a rider as a genetic chromosome: (vehicle, weight of picking up priority, weight of dropping off priority), and a solution to the ridesharing routing problem of arranging vehicles for m riders as:

$$\begin{bmatrix} r_1 : (v^1, w^{1[s]}, w^{1[t]}) \\ r_2 : (v^2, w^{2[s]}, w^{2[t]}) \\ \dots \dots \\ r_m : (v^m, w^{m[s]}, w^{m[t]}) \end{bmatrix} \quad (1)$$

For rider r_i , v^i denotes the vehicle that serves r_i a ride, $w^{i[s]}$ is a positive real number that represents the priority weight of picking up r_i , and $w^{i[t]}$ is a real number that indicates the priority weight of r_i to get off v^i at its destination. A higher priority weight implies a greater sense of urgency to start or finish a ride. Thus, for a vehicle that offers a ride to multiple riders, it will stop at nodes to pick up and drop off the riders with respect to their priority weights. In reality, the priority weight can be the time of a ride request, arriving time or even promotion tips.

2.5. Solution to Route

To calculate the routes for vehicles there are two factors to satisfy: feasibility and uniqueness. Feasibility requires that when a vehicle follows a route to pick up and drop off riders, the number of riders at any given time must not exceed the capacity of the vehicle. Uniqueness requires that, given a solution, one should be able to derive one and only one way to route the vehicles from it. The uniqueness criterion is necessary because it is the solutions that the genetic algorithm evaluates and optimises while objectives in the evaluation and optimisation are based on the routes for vehicles. Thus, to be able to evaluate a solution, we require a 1-1 correspondence with routes that the solution entails. We will explain our method to map a solution to a feasible and unique routing in the following, and introduce the objectives afterwards.

First, to capture the meaning of the priority weights in a solution, we use the following two rules when comparing the priority weights of different riders:

1. No consideration of dropping off priority for riders who are waiting for pick-up.

2. For riders with equal priority weights, the rider with a lower index number is prioritised, considering that the rider posted its ride request earlier.

Algorithm 1: Routing Algorithm:

Input: *Solution*, an array; *V*, vehicles; *R*, ride requests.

Output: *Routes*, an array

```

1 foreach  $v \in V$  do
2   Routes[v]  $\leftarrow \{v[s]\}$ ;
3   Board  $\leftarrow \emptyset$ ;
4   load  $\leftarrow \text{Size}(\text{Board})$ ;
5   Hold  $\leftarrow \text{GetPassengers}(v)$ ;
6   while Board  $\neq \emptyset$  & Hold  $\neq \emptyset$  do
7     rx, wrx  $\leftarrow \text{GetRiderWithGreatestPickupWeight}(\text{Hold})$ ;
8     ry, wry  $\leftarrow \text{GetRiderWithGreatestDropoffWeight}(\text{Board})$ ;
9     if wrx < wry then
10      Board.remove(ry);
11      load = load - 1;
12      Routes[v].add(ry[t]);
13    else
14      if load < v[p] then
15        Hold.remove(rx);
16        Board.add(rx);
17        load = load + 1;
18        Routes[v].add(rx[s]);
19      else
20        Board.remove(ry);
21        load = load - 1;
22        Routes[v].add(ry[t]);
23      end
24    end
25  end
26  Routes[v].add(v[t]);
27 end

```

Regarding a solution, let $\text{Passenger}(v)$ denote the set of riders to whom vehicle v offers a ride, and $\text{Passenger}(v) = \{r_i | v^i = v\}$. The route of v is a sequence of nodes, $\text{Path}(v)$, that are either starting points or destinations of riders in $\text{Passenger}(v)$, and the path that a vehicle travels from a node to another is the shortest path calculated by Dijkstra's algorithm [7]. While a vehicle travels, let $\text{Board}(v, n)$ denote the riders that are on the vehicle v when it visits node n , and $\text{Hold}(v, n)$ be the riders that are still waiting for the vehicle for a pick-up.

Figure 1 and Algorithm 1 illustrate the workflow and steps of our routing algorithm that maps a solution to the routes for a vehicle.⁵ For each vehicle, the very first node in

⁵ For a complete implementation of our routing algorithm, please refer to <https://github.com/Miya-Liu/equitable-ridesharing>.

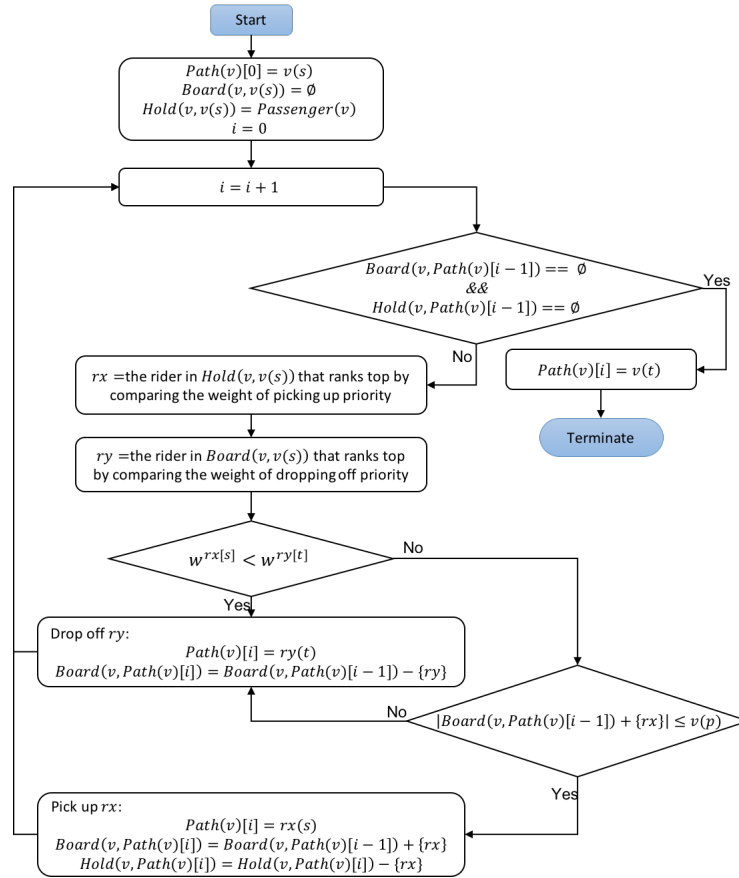


Fig. 1. Algorithm for Mapping Solution to Routes.

its route is its starting point (Line 1), and at that node, the boarding passengers is null (Lines 3-4) and all the riders assigned to it are on hold (Line 5). Then, the next node in the route of the vehicle depends on the priority weights of the aboard and waiting riders. First, we compare the waiting riders' weights of picking up priority and get the top one waiting rider (Line 7), and then compare the aboard riders' weights of dropping off priority and get the top one aboard rider (Line 8). If the aboard rider's weight of dropping off priority is greater than the waiting rider's weight of picking up priority (Line 9), the next node in the route of the vehicle is the destination of the aboard rider (Lines 10-12). Otherwise, we check whether picking up the waiting rider violates the vehicle's capacity constraint (Line 14). If not, the vehicle will travel to the starting point of the waiting rider and pick it up (Lines 15-18). If yes, the vehicle still needs to drop off the aboard rider first (Line 20-22). Until there is no rider aboard or waiting (Line 6-25), the vehicle will travel to its destination (Line 26) and terminates its route.

This routing algorithm guarantees the feasibility of the travel paths of vehicles generated from a solution and the uniqueness of the generation dynamically. Note that, re-

regardless of the uniqueness, it is still possible that different solutions generate the same routes for one or even more vehicles. This is because the different weights of either picking up priority or dropping off priority can result in the same ranking of the riders in the algorithm. Note that the potential redundancies are left to be resolved in the genetic algorithm.

2.6. Objectives

Regarding the travel paths of vehicles and their loaded riders, we introduce and minimise 6 objectives from 3 aspects of sustainability: economic, environmental and social. The economic aspect evaluates the efficiency of the routes generated from a solution with respect to travelling time, waiting time, travel cost and excess distance. Then, the environmental aspect of sustainability considers the impact of vehicles' emission and tries to minimise the total emission of rides. Finally, the social aspect concerns the working time of drivers and aims at reducing the differences among the working time of all drivers.

Economic - Travelling Time (ET): This objective measures the total travelling time of individual riders. The measurement of the travelling time for one rider is the time that it takes from the moment the rider gets on a vehicle until the vehicle drops off the rider at her destination. This includes the time that the vehicle travels and waits to pick up and drop off other riders while the rider is on board. The waiting time of a vehicle includes time periods when the vehicle stops at a starting point of a rider to pick her up. Such a waiting takes place when a vehicle arrive (too) early, i.e., when the arriving time of the vehicle is earlier than the earliest leaving time of the rider. Less travelling time means that the riders entails a more efficient trip.

Let $wait(v, n)$ denote the time that a vehicle v waits for picking riders up at node n along its route. Let $d(n_i, n_j)$ be the shortest distance between node n_i and n_j , and $arrive(v, n)$ represent the time that the vehicle arrives at node n along its route. Hence, $wait(v, n_0) = 0$, $arrive(v, n_0) = 0$, and

$$arrive(v, n_i) = arrive(v, n_{i-1}) + wait(v, n_{i-1}) + \frac{d(n_{i-1}, n_i)}{speed},$$

where $n_i = Path(v)[i]$, and $speed$ is a given average speed. Assume that v will pick up k riders at n_i , thus, $wait(v, n_i) = \max\{0, r_x(u) - arrive(v, n_i)\}$, where $r_x(u)$ is the greatest earliest leaving time among the k riders. Therefore,

$$ET = \sum_r (arrive(v^r, r(t)) - arrive(v^r, r(s))). \quad (2)$$

Economic - Waiting Time (EW): This objective is to evaluate the waiting time for all riders before vehicles pick them up with respect to their earliest leaving time.

$$EW = \sum_r \max\{0, arrive(v^r, r(s)) - r(u)\} \quad (3)$$

Economic - Excess Distance (ED): This objective measures the extra distance that a vehicle travels when it needs to pick up and drop off riders compared to the distance of directly driving from its starting point to the destination. Let n_i be the i th node in a route,

$$ED = \sum_v \left(\sum_{i=0}^{|Path(v)|-1} d(n_i, n_{i+1}) - d(v(s), v(t)) \right) \quad (4)$$

Economic - Travel Cost (EC): This objective measures the cost of all rides in total.

$$EC = \sum_v \left(v(c) \times \sum_{i=0}^{|Path(v)|-1} d(n_i, n_{i+1}) \right) \quad (5)$$

Environmental - Emission (SE): This objective is designed to measure the emission of all the vehicles. By minimising this objective, sharing a ride can reduce pollution. Recall that the emission rate of a vehicle is related to the number of passengers on the vehicle. Therefore, the emission of all the vehicles regarding one solution is

$$SE = \sum_v \sum_{i=0}^{|Path(v)|-1} v(e)[l(v, n_i)] \times d(n_i, n_{i+1}). \quad (6)$$

where $l(v, n) = |Board(v, n)|$.

Social - Working Time (SW): The working time is calculated from the moment a vehicle leaves its starting point until it arrives at its destination, which is $arrive(v, v(t))$. This objective demonstrates the workload of a vehicle. Regarding the social sustainability, this work tries to balance the workload among all drivers and ensure a sustainable ridesharing service that is fairness-aware. Hence, this objective is defined as the Gini coefficient [5] of all vehicles' working time, $W = \{arrive(v_1, v_1(t)), arrive(v_2, v_2(t)), \dots, arrive(v_m, v_m(t))\}$ as follow.

$$SW = Gini(W). \quad (7)$$

With the above-defined multiobjectives, we will later explain our algorithm that generates multiple routing options that balance the six objectives of all three pillars of sustainability.

3. NSGA3 for Sustainable Ridesharing

This work adopts an existing genetic algorithm called Non-dominated Sorting Genetic Algorithm 3 (NSGA3) [13] for dynamic routing in the sustainable ridesharing problem. Figure 2 shows the workflow of the NSGA3 with modifications for the ridesharing setting.

NSGA3 requires no configurations of the importance or weights of multiple objectives in the optimisation, but balance them automatically. The optimisation procedure includes:

1. **Sampling**: Given the number of riders, vehicles and sample size, it generates an initial sample population. In this step, for each solution in the population, we randomly assign vehicles to serve riders, and assign random values as the weights of the picking up and dropping off priority of riders.

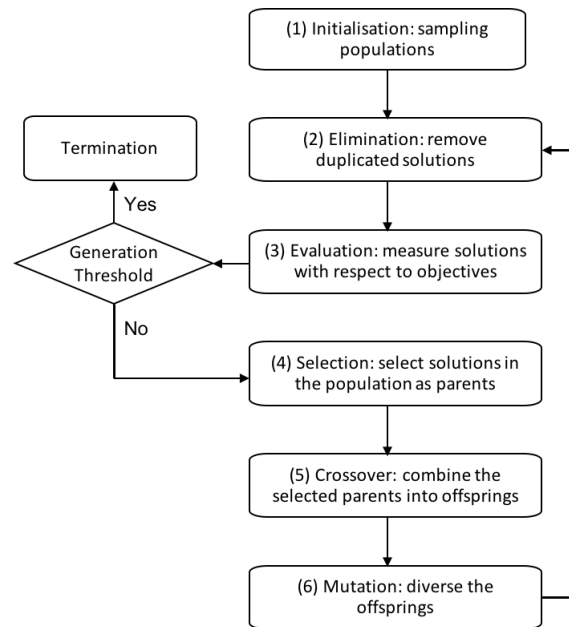


Fig. 2. Modified NSGA3 for Sustainable Ridesharing.

2. Elimination: It deletes duplicate solutions in population. And if the size of the current population after elimination is smaller than the initial population, the following introduced crossover process is repeated until the desired number of offspring is fulfilled.
3. Evaluation: For individual solution, it calls the routing algorithm first for a feasible ridesharing arrangement of the solution and evaluates the solutions according to the predefined objectives or any other customised metrics or constraints.
4. Selection: It selects some solutions as parents for generating offspring in next generation. NSGA3 uses a reference points [4] based selection operator. As we applying this genetic algorithm for multiobjective optimisation, this selection is ideal as it is guided by specifying a set of well-maintain diversity in the population regarding different objectives.
5. Crossover: It combines the selected parent solutions to generate offspring solutions. We define the crossover as two parent solutions generating one offspring. The pattern to generate an offspring for each pair of parents is to use the first half chromosomes from a parent and the second half chromosomes from the other parent to generate an offspring solution. Note that our implementation supports splitting both parents into any number of slices and then selecting the same number of slices to generate an offspring.
6. Mutation: It mutates offspring to increase the diversity of the current population. The modified mutation is: for each offspring, we select half riders and change the value of its corresponding chromosomes by (1) changing the vehicle assigned to a rider; (2) increasing its weight of picking up priority by a positive number; (3) increasing its weight of dropping off priority by a random positive number.

Table 2. Information about Group Instances.

Parameters	Variation in Instances
# Riders	[53, 48, 24, 16, 16]
Riders' Location	[Original, Noise in s , Noise in t , Noise in Journey]
# Vehicles	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Vehicle Type	[1, 3, 6, 10]

Table 3. Algorithm Parameter Settings

Parameters	Value/Range
Population	10
Offset	5
Generation	[100, 200, 300, ..., 1000]
Objectives	[EW, ET, ED, EC, SE, SW]

Generally speaking, the parameters, including the size of population, number of parents, crossover method, mutation method, number of generations and termination threshold, would affect the performance of the genetic algorithm. After observing the impact of those parameters on small test data, we choose the above-mentioned parameters in our algorithm settings, and the threshold of the number of generations as the condition to terminate the optimisation while attempted to vary the threshold in a wide range to reduce the impact of other static parameters.

This approach will automatically generate multiple routing solutions when we set the size of population greater than 1. In addition, the routing solutions are feasible and balance the economic, environmental and social sustainability, while the algorithm optimises the six objectives. Note that we did not consider the objectives when calculating the shortest path among all nodes of a map graph. This is because the objectives are defined and calculated based on the paths of all the vehicles that they will drive, pick up, and drop off riders. For instance, we cannot get the override distance just for the edge between two nodes.

4. Experimental Evaluation

The main goal of this work is to demonstrate the impact of sustainable objectives in ridesharing routing and the efficiency of our GA-based routing approach. We present 5 groups of experiments with various numbers of riders, vehicles, objectives and generations to illustrate the effectiveness of our approach. This section evaluates the performance with respect to the standard metrics in the field of vehicle routing [14,3] and includes social and environmental metrics such as the waiting time of a rider to start a ride.

4.1. Data Sources

We use the Cargo benchmark dataset [18], which takes data from the ridesharing company Didi. The instances have maximum 65,500 riders and 50,000 vehicles over a long time horizon and a scale of 876km² area. Since we focus on one-shot routing, we take slices from the dataset for our evaluation. Note that in practice, new routes can be calculated as more requests come in.

- Road Map: We use the road map of Manhattan from Cargo [18]. It has 12,320 nodes, 15,722 edges in an area of 59km².
- Instances: We design 5 groups of riders, and experiments with instances as detailed in Table 2. The variation column shows the range of parameters that we vary in the experiments.

- **Riders:** For each group, there is a fixed number of riders. To discover the possibility of riders to share a vehicle, we first construct a graph using the starting points and destination of original 5,033 riders as nodes and their journeys as edges, and then select a set of nodes whose clustering coefficient is not zero. With the selected node set, we extract riders whose starting points or destination fall into the set. Last, we put those riders into 5 clusters by K-means clustering method, which are the original riders in each group of experiments. In addition, we generate another 3 variations of the original riders by adding noises to either the riders' starting points, or destination, or both of them. The noise is added by randomly change the target nodes to one of its neighbors on the road map.
- **Vehicles:** For each group, the number of vehicles range from 1 to 10 by 1 and location of vehicles are randomly picked from Cargo. At each fixed number of vehicles, we allow all vehicles to be either one type of vehicles from Table 1.

With above constructed instances, we can examine the impact of number and location of riders, and number and type of vehicles on the performance of ridesharing routing, by varying the above parameters in the experiments.

4.2. Benchmark - Greedy Routing

An greedy-based routing algorithm is designed and implemented as the baseline for ridesharing performance evaluation. For a group of riders, the greedy routing algorithm first sorts all ride requests by their posting time. Then, for each ride request in the rank, it searches over all the vehicles to find one that is closet to the starting point of the request at its early leaving time. The selection is based on the distance from the current location of a vehicle, to its travelling destination if it has passenger on board and to the starting point of the request. After that, the selected vehicle will travel to the starting point of the request and directly drive the rider to its destination. Note that the algorithm will not change the arrangement made before receiving the request of new riders. At the end, we have the routes of all vehicles and evaluate the routes with respect to the same objectives used in our algorithm.

In the following discussion, all the displayed results regarding the six objectives is the relative reduction in comparison with the ones of greedy routing algorithm.

4.3. Experiment Setting

The implementation of the GA algorithm in the Python programming language (using pymoo⁶) allows configuring the population size, number of generations, the offspring rate, and muting/enabling different objectives. Table 3 displays the settings of algorithm parameters. To evaluate our approach, we set the population size and offspring size to 10 and 5, respectively. In the experiments, the generation varies from 100 to 100 by 100 to briefly demonstrate the effects of large generations. Additionally, to evaluate the effectiveness of our approach with respect to the 6 different objectives from 3 aspects, we set

⁶ <https://pymoo.org/>

up experiments with either one objective or the other left 5 objectives to compare the optimised routes with those after optimising the all 6 objectives.⁷ Therefore, in the rest of this section, we will illustrate our experiment results and answer the questions: How the generation affects the performance, how the number of vehicles affects the performance and how the setting of objectives affects the performance and the sharing rate of each vehicle.

4.4. Experiment Results

This section examines and illustrates our experiment results regarding the sharing rate, optimisation and balance of objectives and diversity among the generated ridesharing routes. The sharing rate measures the efficiency of the ridesharing routing algorithm in promoting multiple riders in one vehicles during their trips. With respect to the 6 different objectives, we will compare the efficiency of the ridesharing routing algorithm in optimising each individual objectives against the all objectives and their balance, in order to have a further view of the relationship among the 6 different objectives. At the end, we evaluate the diversity of the generated solutions by the GA-based ridesharing routing algorithms which can be an ideal voting pool for what we call participatory route selection (see Section 5).

Sharing Rate To measure the efficiency of our routing algorithm in promoting ride sharing, we define a sharing rate metric, Sharing Rate Over Time (SROT), as follow:

$$SROT = \frac{|Path(v)|}{\sum_{i=0}^{|Path(v)|} |Board(v, Path(v)[i])| * d(Path(v)[i], Path(v)[i + 1]) / speed.}$$

For one vehicle, this metric calculates the average number of riders who share the vehicle along its complete travelling path.

Figure 3 uses boxplot to illustrate the sharing rate of the generated routes for 53 riders, 5 vehicles with varied capacities while setting up the 6 objectives and 1000 generations. The 4 subplots correspond to the 4 variations of riders' location. Comparing to the results of vehicles with a capacity of 1, the sharing rate of greater capacities is higher. In total, the capacity of 3, 6, 10 increase the sharing rate by 61.63%, 96.35% and 110.19% on average, respectively. In addition, comparing the mean sharing rate of 4 different vehicle capacities when the riders' locations vary, the vehicles with capacity of 6 outperform the others in Figure 3(a) and (c), while capacities of 3 and 10 have greatest mean sharing rate in Figure 3(b) and (d), respectively. The standard deviation of the 4 different capacities when the locations of riders vary, can represent the impact of riders' locations on the ridesharing efficiency of our routing algorithm, which are 0.0096, 0.0493, 0.1255 and 0.1928. This infers that when the capacity of vehicles increases, their sharing rates are more likely to be affected by the locations of riders.

Furthermore, Figure 4 shows the average sharing rate of vehicles when the number of riders and vehicles changes. The darker area in the bottom right corner of Figure 4

⁷ To comply with the anonymity requirements of the track, we excluded the link to the repository in this version. The instance and algorithm implementation files will be provided in the next version. For the complete experiment results please refer to the support material.

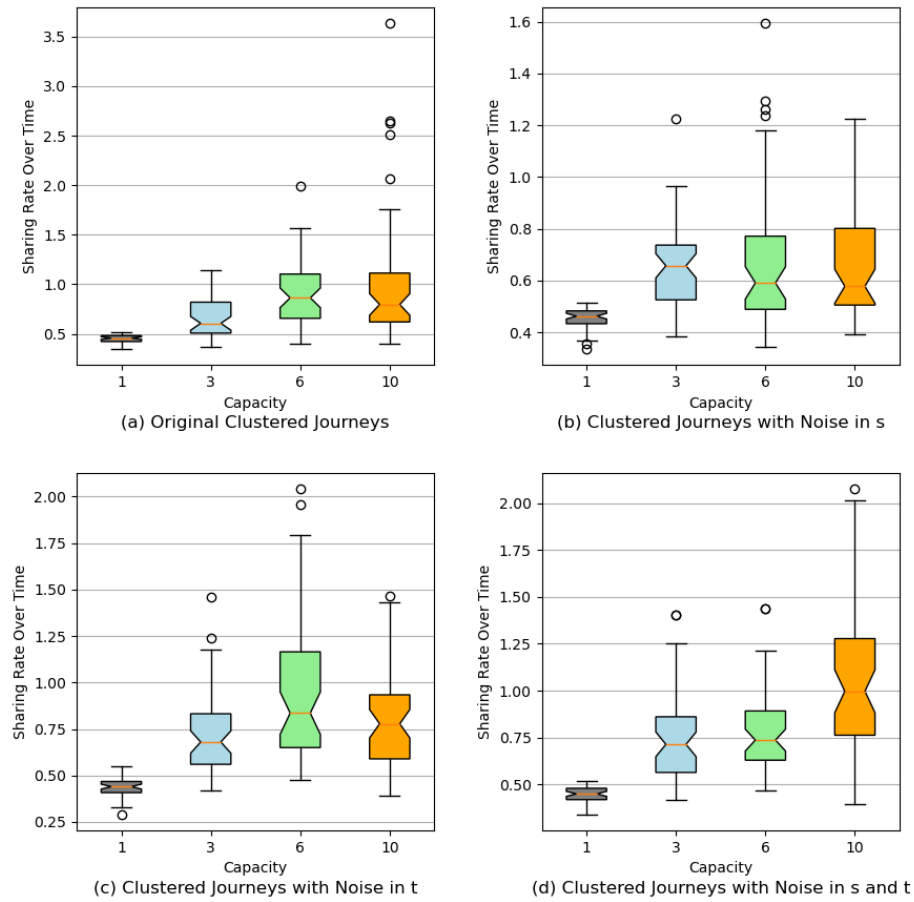


Fig. 3. Sharing Rate of Arranged Routes for Riders in Group 1 with 5 Vehicles after Optimising 1000 Generations for 6 Objectives

marks the sharing rate lower than 0.5. In these area, capacities of all available vehicles are greater than the number of riders, which indicates there is not enough riders to share vehicles. Considering the optimising objectives, SW and EW, which aim to balance the working hour of all vehicles and reduce the waiting time of riders, the sharing rate of vehicles will reduce when the number of riders decrease to be smaller than the overall capacities of vehicles. The brighter cells in Figure 4 are in the top and left corner when there are more riders for the algorithm to arrange ridesharing routes for vehicles.

To summarise, when the capacity of vehicles increases, the sharing rate will increase on average. However, it does not imply higher capacity will always result in greater sharing rate. First, sharing rate is related to the location of riders. When riders are randomly located at their starting point but are travelling to the same direction, vehicles with a capacity of 3 would gain better sharing rate. When riders start their journeys from close

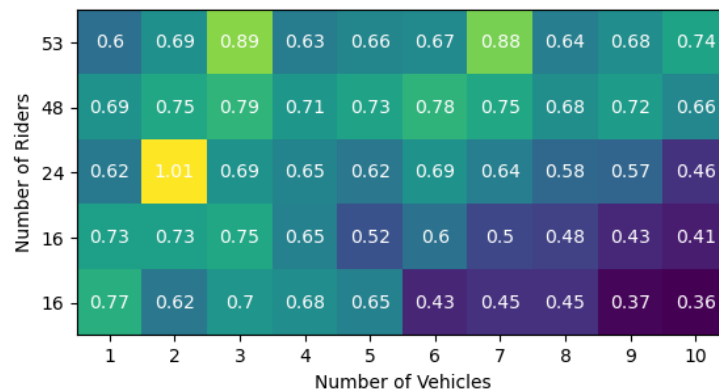


Fig. 4. Average Sharing Rate of Vehicles with Capacity of 3

location but travel to different destinations, the vehicles with a capacity of 6 performs better. In addition, the number of riders and vehicles have impact on the sharing rate of vehicles. When the number of riders is far greater than the capacities of all vehicles, the sharing rate of vehicles is more likely to be higher. And knowing the number of riders is small, sending out less vehicles with smaller capacity can help increase their sharing rate.

Objectives To evaluate the performance of our routing algorithm with respect to individual objectives, we use the greedy routing algorithm as the baseline. Figure 5 shows the percentage reduction of the generated solutions to arrange routes for 53 riders, 3 vehicles with capacity of 3. Each sub figure displays the percentage reduction of individual objectives when optimising all objectives (All-On), one objective (EW/ET/EC/ED/SE/SG On) and the other objectives (EW/ET/EC/ED/SE/SG Off). Throughout all charts in Figure 5, we can find that all objectives are not independent. For instance, as Figure 5e shows, when only minimising emission, the routing solution turns out to have the better performance regarding the objective EW and ET. In addition, the switch optimisation results of ET and SW affect each other the most, as Figure 5b and 5f showed, i.e. when social inequality improves most the travelling time increases most and vice versa.

Moreover, the percentage reductions at objective ED and EC are always at the same pace over the six group of comparison. And our ridesharing algorithm outperforms greedy routing in optimising ED and EC by 8.9% while optimising all objectives, 16.9% if only optimising those two, and at least 5.2% in other cases. The ridesharing routes generated by our algorithm constantly require higher results in EW and ET, because greedy routing always try to let each rider get on a vehicle as quick as possible and directly drive individual riders from their starting points to destinations, which results in minimum ET and the better performance of reducing EW. As to SW, the ridesharing routes contribute to unbalanced working hours among vehicles excepting when only optimising SW and improved it by 65.7%. Hence, the ridesharing algorithm is capable to optimising individual objectives as well as dependent multiple objectives.

Notably, the improved percentage over greedy regarding the 6 objectives falls in varied scales, especially over -100% of EW in 5a and 16.9% of ED in 5d. The reason is that the

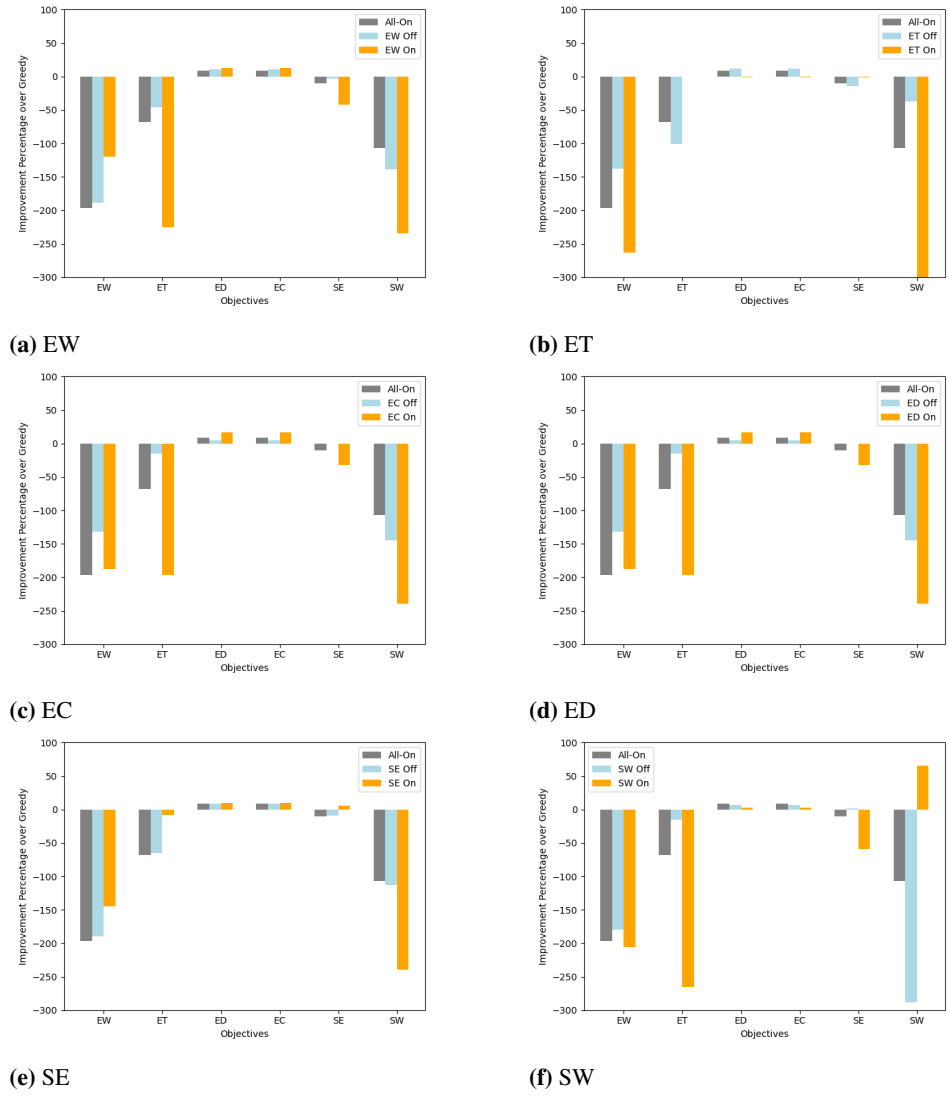


Fig. 5. Percentage Reduction in Comparison of Optimising One Objective, the Other Objectives and All Objectives with 53 Riders, 3 Vehicles with Capacity of 3

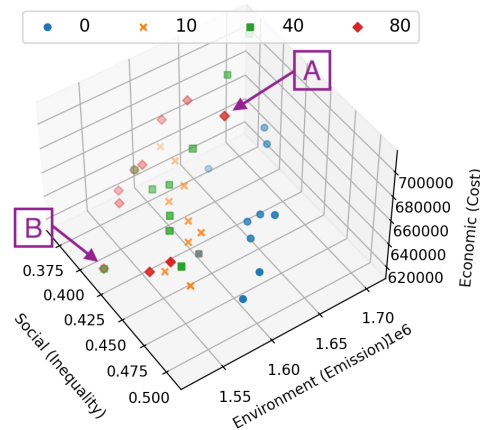


Fig. 6. Evolution of GA Solutions

primer task of ridesharing to fulfil is to drive all riders from their distinct starting points to their destinations, leaving limited space for improvements of ED, EC and SE. The public transportation, such as underground, that has numeric capacity and riders get on and off it at stable stops, are believed to be more economic and environment friendly. In our future work, public transportation will be considered in ridesharing, and our approach will try to deliver suggested routes involve public transportation with an estimated reduction of economic cost and emission.

To further understand the balancedness of ridesharing routing solutions with respect to the economic, environmental and social aspects of sustainability, we plot the results for 53 riders and 10 vehicles with capacities of 3 in Figure 7. In general, as the optimisation proceeds, the environmental and economic sustainability improves while the social inequality fluctuates between -1.5 and -2.5. In addition, we plotted the routing results in 4 sampling generations for 150 riders and 100 vehicles with capacity of 1 aiming at optimising 6 objectives. Figure 6 presents the evolution of the populations at generations 0, 10, 40 and 80. The solutions in each population have diverse effectiveness against the three aspects of sustainability. Basically, as the optimisation proceeds, the social inequality decreases. The initial generation (in blue) has greater social inequality than other generations while the 80th generation has the lowest inequality. However, from the perspective of economics and environment, the costs and emissions do not improve for all solutions. Although, the ridesharing routing algorithm can balance and improve the overall objectives, but optimising SW separately would result in lower social inequality.

Furthermore, it is observable that we have a more diverse set of solutions. For instance, solution *A* (in the 80th generation) has a low economic cost and social inequality which compensates for its high environmental emission level. The other notable solution is labelled with a boxed *B* which performs well against all the three dimensions. We will further discussion the solution diversity in the following section.

Solution Diversity To understand and illustrate the diversity of populations, we plot the 10 solutions to routing 16 riders and 10 vehicles with capacity of 3 in Figure 8. It

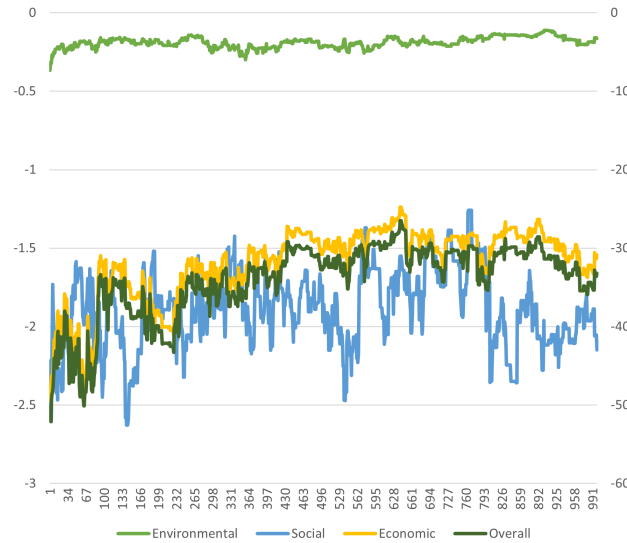


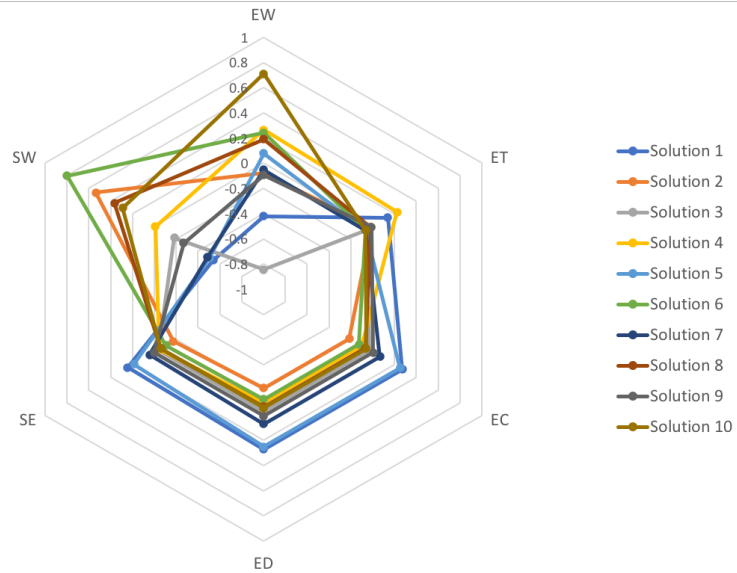
Fig. 7. Performance on Optimising Objectives over Generations

presents the radar plot of effectiveness of each solution with respect to the 6 objectives, after normalising the results by the mean value of 10 solutions. The smaller number of an objective implies a better performance of a solution on that objective. Among these 10 solutions, the effectiveness regarding social sustainability and riders' waiting time vary greater than the others. This is because the changes in allocating riders from a vehicle to another directly affects the working time of the vehicles and their waiting time. The population offers diverse solutions that improve social inequality between 0.13 to 0.52 whereas riders' waiting time between 2768 to 254. We expect to take advantage of populations' diversity in the next phase of our work on sustainable mobility and allow riders to vote on routes with respect to their concerns, such as economic costs, environment emissions, or social inequality.

5. Conclusions

In this work, we presented a multiobjective evolutionary approach based on GA algorithm for generating routing options in sustainable ridesharing. Although there are well-studied multiobjective optimisation methods [12], a GA algorithm generates a diverse set of solutions naturally for the ridesharing problem. Our method is not only sustainability-aware but also establishes a foundation for explainable, participatory, and dynamic ridesharing services.

Explainability for Riders, Drivers, and Operators: In comparison to data-driven techniques with black-box optimisation components, in our approach, stakeholders can be provided with visualisations to see how different objectives (e.g., minimising emissions) affect routing solutions. For instance, they can be presented using graphs as in Figure 5



Solution	EW	ET	EC	ED	SE	SW
1	940	25026.50	488337	162779	59578.75	0.13
2	1491	21725	301746	100582	39623.52	0.44
3	254	20593	371610	123870	46123.18	0.23
4	2051	26893	345294	115098	45876.63	0.28
5	1749	20593	480135	160045	56762.88	0.14
6	2009	20593	335763	111921	42608.76	0.52
7	1535	20593	410178	136726	49815.50	0.15
8	1930	21545	358578	119526	44954.98	0.39
9	1470	21725	386103	128701	47976.06	0.21
10	2768	20593	361275	120425	45021.09	0.37
Avg	1619.70	21987.95	383901.90	127967.30	47834.14	0.29
Max	2768	26893	488337	162779	59578.75	0.52
Min	254	20593	301746	100582	39623.52	0.13

Fig. 8. Diversity of 10 Solutions w.r.t. Each Objective

and with explanations on how waiting a bit more (in comparison to using private rides) can benefit the environment or the fairness of the service for drivers.

Promoted sustainability: With the awareness of sustainability, our approach provides a feasible solution to promote ridesharing with respect to varied type of vehicles, and highlights the factors associated with the sharing rate when promoting ridesharing is possible. We demonstrate that the average sharing rate among riders can be improved when the capacity of vehicles increases, although higher capacity can not always result in greater sharing rate. In addition, the locations and numbers of riders also affect the sharing rate the number of riders and vehicles have impact on the sharing rate of vehicles. For numerous riders starting from close locations, vehicles with a greater capacity would contribute to a higher sharing rate, where riders from random locations but heading to close destinations can share vehicles with a lower capacity for a higher sharing rate.

Participatory Route Selection: Building on this approach, ridesharing operators can present routing options to riders (or autonomous agents that represent riders) and allow voting among them. This way, users can directly participate in the route selection process and opt for the most collectively equitable route. Our diverse set of GA solutions are not ranked. Thus, a set of riders may prefer one over another and to allow that, we aim to extend our work by adding a preference/vote-based route ranking module in the future.

Dynamic Fine-Tuning: Our approach allows dynamic fine-tuning over time. Users and service operators can inspect routing solutions, evaluate if they are realistic and feasible, and participate in fine-tuning the route generation algorithm and the objective weights to set trade-offs. One can use focus groups for such a tuning over time—e.g., as a city and its citizens change—to enable dynamic fine-tuning of sustainable ridesharing services.

We aim to extend our work by integrating a participatory route selection process and allowing users to vote over a diverse set of routing solutions with all the objectives and then also muting one or two objectives to provide solutions that match diversity in users' preferences. With a better understanding of users' preferences, we aim to explore other methods for multiobjective optimisation in the context of ridesharing service. For example, we can define the assignment of a rider to a vehicle as a move and evaluate the move with respect to the multiple objectives, and then adopt reinforcement learning for this problem. Moreover, we plan to test the efficacy of our approach in larger datasets and investigate simulation-based methods to analyse how different map structure and spatio-temporal properties of requests affect the optimality and equitability of solutions.

Data access statement. This study was a reanalysis of data that are publicly available from the the Cargo benchmark dataset [18]. Implementations and data derived through the reanalysis undertaken in this study are available from the public GitHub repository at <https://github.com/Miya-Liu/equitable-ridesharing>.

Acknowledgments. We thank the anonymous reviewers and participants at the ATT 2022 Special Section of Computer Science and Information Systems (ComSIS) Journal, for their incisive comments that were most useful in revising this paper. This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Fellowship (EP/V022067/1) on Citizen-Centric AI Systems (<https://ccaais.ac.uk/>) and the platform grant entitled “AutoTrust: Designing a Human-Centred Trusted, Secure, Intelligent and Usable Internet of Vehicles” (EP/R029563/1). For the purpose of open access, the author has applied a creative commons attribution (CC BY) licence to any author accepted manuscript version arising.

References

1. Atasoy, B., Ikeda, T., Song, X., Ben-Akiva, M.E.: The concept and impact analysis of a flexible mobility on demand system. *Transportation Research Part C: Emerging Technologies* 56, 373–392 (2015)
2. Bardaka, E., Hajibabai, L., Singh, M.P.: Reimagining ride sharing: Efficient, equitable, sustainable public microtransit. *IEEE Internet Computing* 24(5), 38–44 (2020)
3. Ben Cheikh-Graiet, S., Dotoli, M., Hammadi, S.: A tabu search based metaheuristic for dynamic carpooling optimization. *Computers & Industrial Engineering* 140, 106217 (2020), <https://www.sciencedirect.com/science/article/pii/S0360835219306862>
4. Blank, J., Deb, K., Roy, P.C.: Investigating the normalization procedure of nsga-iii. In: *EMO*. pp. 229–240 (2019)
5. Catalano, M.T., Leise, T.L., Pfaff, T.J.: Measuring resource inequality: The gini coefficient. *numeracy* 2 (2): Article 4 (2009)
6. Colorado Department of Transportation: Overview of transit vehicles (2022), https://www.codot.gov/programs/innovativemobility/assets/commuterchoices/documents/trandir_transit.pdf
7. Cormen, T.H.: Section 24.3: Dijkstra’s algorithm. *Introduction to algorithms* pp. 595–601 (2001)
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation* 6(2), 182–197 (2002)
9. EMISIA SA Corporate: Copert the industry standard emissions calculator (2018), <https://www.emisia.com/utilities/copert/>
10. Fonseca, C.M., Fleming, P.J.: Multiobjective genetic algorithms. In: *IEE colloquium on genetic algorithms for control systems engineering*. pp. 6–1. Iet (1993)
11. Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.E., Wang, X., Koenig, S.: Ridesharing: The state-of-the-art and future directions. *Transportation Research Part B: Methodological* 57, 28–46 (2013)
12. Hayes, C.F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., et al.: A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems* 36(1), 1–59 (2022)
13. Jain, H., Deb, K.: An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach. *IEEE Transactions on evolutionary computation* 18(4), 602–622 (2013)
14. Laporte, G.: Fifty years of vehicle routing. *Transportation Science* 43(4), 408–416 (nov 2009)
15. Liu, M., Yazdanpanah, V., Stein, S., Gerding, E.: Multiobjective routing in sustainable mobility-on-demand. In: *ATT’22: Workshop Agents in Traffic and Transportation*, July 25, 2022, Vienna, Austria: Part of IJCAI 2022 (23/07/22 - 29/07/22). pp. 47–61 (July 2022)
16. Millen, S., Page, E.: Transport and environment statistics 2021 annual report (2021), https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/984685/transport-and-environment-statistics-2021.pdf
17. Murata, T., Ishibuchi, H., et al.: Moga: multi-objective genetic algorithms. In: *IEEE international conference on evolutionary computation*. vol. 1, pp. 289–294. IEEE Piscataway, NJ, USA (1995)
18. Pan, J.J., Li, G., Hu, J.: Ridesharing: simulator, benchmark, and evaluation. *Proceedings of the VLDB Endowment* 12(10), 1085–1098 (2019)
19. Purvis, B., Mao, Y., Robinson, D.: Three pillars of sustainability: in search of conceptual origins. *Sustainability science* 14(3), 681–695 (2019)

20. Ramchurn, S.D., Stein, S., Jennings, N.R.: Trustworthy human-ai partnerships. *iScience* 24(8), 102891 (2021)
21. Simonin, G., O’Sullivan, B.: Optimisation for the ride-sharing problem: a complexity-based approach. In: *ECAI 2014*, pp. 831–836. IOS Press (2014)
22. UK Government: Driving licence categories (2022), <https://www.gov.uk/driving-licence-categories>
23. UK NAEI - National Atmospheric Emissions Inventory: Primary NO2 emission factors for road vehicles (2022), https://naei.beis.gov.uk/resources/Primary_NO2_Emission_Factors_for_Road_Vehicles_NAEI_Base_2021_v3.pdf
24. Weigel, B.A., Southworth, F., Meyer, M.D.: Calculators to estimate greenhouse gas emissions from public transit vehicles. *Transportation research record* 2143(1), 125–133 (2010)
25. Yu, Y., Wu, Y., Wang, J.: Bi-objective green ride-sharing problem: Model and exact method. *International Journal of Production Economics* 208, 472–482 (2019)

A. NSGA3 Implementation

A.1. Sampling

Algorithm 2: Sampling(n_riders , $n_vehicles$, $n_samples$)

```

1 Population  $\leftarrow \emptyset$ ;
2 for  $1 \leq s \leq n\_samples$  do
3   solution  $\leftarrow \emptyset$ ;
4   for  $1 \leq i \leq n\_riders$  do
5     chore['vehicle']  $\leftarrow$  random( $n\_vehicles$ );
6     chore['pickup_weight']  $\leftarrow$  random( $100 * n\_riders$ );
7     chore['dropoff_weight']  $\leftarrow$  random( $100 * n\_riders$ );
8     solution.append(chore);
9   end
10  Population.append(solution);
11 end
12 return Population;
```

A.2. Elimination

Algorithm 3: Elimination(Population, $n_samples$)

```

1 foreach  $s \in Population$  do
2   foreach  $s' \in Population - \{s\}$  do
3     if  $s$  equals to  $s'$  then
4       Population  $\leftarrow$  Population -  $\{s'\}$ ;
5     end
6   end
7 end
8 if Population.size <  $n\_samples$  then
9   Population  $\leftarrow$  Crossover();
10 end
11 return Population;
```

A.3. Evaluation

Algorithm 4: Evaluation(V , $solution$, obj_list)

```

1 basic_objs ← [EW,ET,EC,ED,SE,SW];
2 Routes ← Routing(solution);
3 res ←;
4 foreach  $obj \in basic\_objs$  do
5   | if  $obj \in obj\_list$  then
6   |   | res.append(Cal(obj, Routes))
7   | end
8 end
9 return res;

```

A.4. Selection

Algorithm 5: Selection($Population$, $n_selection$, $n_parents$)

```

1 count ←  $n\_selection \times n\_parents$ ;
2 multi ←  $\lceil \frac{count}{Population.size} \rceil$ ;
3 selection ← pymoo.util.misc.random_permutations(multi,
   Population.size)[0: count];
4 return selection;

```

A.5. Crossover

Algorithm 6: Crossover(Parents, $n_offspring$, n_slices)

```

1 parent_A ← Parents[1];
2 parent_B ← Parents[2];
3 offsprings ← ∅;
4 for  $1 \leq o \leq n\_offspring$  do
5   one_offspring ← ∅;
6   num ← parent_A[o].size ÷ n_slices + 1;
7   for  $1 \leq j \leq num$  do
8     from_slice ← num × j;
9     to_slice = num × j + num;
10    if to_slice > parent_A[o].size then
11      | to_slice ← parent_A[o].size;
12    end
13    if  $j \bmod 2 = 0$  then
14      | one_offspring.concat(parent_A[o][from_p:to_p]);
15    else
16      | one_offspring.concat(parent_B[o][from_p:to_p]);
17    end
18  end
19  offsprings.append(one_offspring);
20 end
21 return offsprings;
```

A.6. Mutation

Algorithm 7: Mutation(One.Offspring)

```

1 mutation_items ← [1 to One.Offspring.size/2];
2 mutated_offspring ← ∅;
3 foreach  $i \in mutation\_items$  do
4   item ← One.Offspring[i];
5   item[i]['vehicle'] ← random( $n\_vehicles$ );
6   item[i]['pickup_weight'] ← random( $10 * n\_riders$ );
7   item[i]['dropoff_weight'] ← random( $10 * n\_riders$ ) + 1;
8   mutated_offspring.append(item);
9 end
10 return mutated_offspring;
```

Mengya Liu is a Staff Research in the Smart Data group in the lab of Artificial Intelligent at Lenovo Research of Lenovo (Beijing) LTD. Her research focuses on optimisation algorithms, decision-making technologies and graph neural networks, and its application areas are supply chain transformation and B2B pricing.

Vahid Yazdanpanah is an Assistant Professor of Computer Science in the Agents, Interaction and Complexity (AIC) research group in the School of Electronics and Computer Science at the University of Southampton. Vahid's main focus is on intelligent agent technologies, decision support concepts, and formal methods for multiagent systems.

Sebastian Stein is a Turing AI Fellow and Professor in the Agents, Interaction and Complexity (AIC) research group in the School of Electronics and Computer Science at the University of Southampton. He works on citizen-centric AI systems and incentives in multi-agent systems, with a focus on smart transportation and smart energy.

Enrico Gerding is a Professor and head of the Agents, Interaction and Complexity research group in the School of Electronics and Computer Science at the University of Southampton. His research focuses on multi-agent systems, specifically game theory and mechanism design, and its application to areas such as transport, maritime and energy.

Received: December 09, 2022; Accepted: July 04, 2023.

Knowledge Transfer in Multi-Objective Multi-Agent Reinforcement Learning via Generalized Policy Improvement

Vicente N. de Almeida, Lucas N. Alegre, and Ana L. C. Bazzan

Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS)
Porto Alegre, RS, Brazil
{vnalmeida,lnalegre,bazzan}@inf.ufrgs.br

Abstract. Even though many real-world problems are inherently distributed and multi-objective, most of the reinforcement learning (RL) literature deals with single agents and single objectives. While some of these problems can be solved using a single-agent single-objective RL solution (e.g., by specifying preferences over objectives), there are robustness issues, as well the fact that preferences may change over time, or it might not even be possible to set such preferences. Therefore, a need arises for a way to train multiple agents for any given preference distribution over the objectives. This work thus proposes a multi-objective multi-agent reinforcement learning (MOMARL) method in which agents build a shared set of policies during training, in a decentralized way, and then combine these policies using a generalization of policy improvement and policy evaluation (fundamental operations of RL algorithms) to generate effective behaviors for any possible preference distribution, without requiring any additional training. This method is applied to two different application scenarios: a multi-agent extension of a domain commonly used in the related literature, and traffic signal control, which is more complex, inherently distributed and multi-objective (the flow of both vehicles and pedestrians are considered). Results show that the approach is able to effectively and efficiently generate behaviors for the agents, given any preference over the objectives.

Keywords: reinforcement learning, multi-agent systems, multi-objective decision making, generalized policy improvement, traffic signal control.

1. Introduction

Reinforcement Learning (RL) [33] deals with agents that learn by acting in an environment and receiving rewards (numerical signals) that guide them toward selecting better actions. In recent years, RL has been successfully applied to complex tasks, such as healthcare [40], robotics [20], game playing [32, 24] and combinatorial optimization [23].

While multi-agent RL (MARL) is a natural framework to model problems that are inherently distributed, such formulation adds many challenges to RL. For instance, because multiple agents interact in a shared environment, their actions are usually highly coupled. This makes learning harder as agents try to adapt to other agents that are also learning. Besides, several convergence guarantees no longer hold when agents learn in a decentralized manner [9]. However, in many real-world problems, where the control is decentralized, it is not always possible, feasible or desirable to avoid a MARL formulation.

Besides considering multiple agents, RL has also been extended to deal with multiple objectives in the multi-objective RL (MORL) literature [17]. This is of fundamental practical importance, since many real-world problems are inherently multi-objective (e.g. in traffic signal control we may need to trade-off the waiting time of the vehicles and the pedestrians). There are two main ways to formulate a MORL problem: using separate reward functions for each objective, or using a single scalar reward that combines the agent's preferences with the values received for each objective, thus reducing a MORL problem into a single-objective RL problem. While the literature argues for both these formulations, in [29] many scenarios are presented to illustrate the need for having methods that deal with separate reward functions to solve MORL problems. For instance, the user may not know a priori what is the more appropriate trade-off among the objectives, or the underlying user preferences may change over time. To deal with such settings, the goal is to construct a set of policies such that agents can perform well, given *any* preferences or ways of combining the objectives. In this paper, we make the assumption that the preferences over objectives are expressed as linear combinations of the reward functions.

Recently, Alegre et al. [3] introduced a method that tackles this problem and enables an RL agent to transfer knowledge from policies specialized to different objectives. Their method incrementally constructs a set of policies that can later be combined to generate optimal policies for any preference among objectives. However, their proposed method only addresses single-agent scenarios. To fill this gap, we extend the method presented in [3] in order to consider multiple agents. Our approach builds a shared set of policies in a decentralized way during training, and then combines these policies to create new policies specialized to different preferences over the objectives. The proposed method is applied to two different environments: a multi-agent extension of the Four-Room environment (a domain commonly used in the related literature), and a complex, inherently distributed and multi-objective problem (traffic signal control, where objectives relate to the waiting time of both vehicles and pedestrians).

In summary, the present method works as follows. During the training phase, agents are iteratively given different preferences over their objectives, and learn policies based on these preferences. These policies are shared among all agents, which have the same observation and action spaces. During the execution phase (after the training is over), by making use of generalized policy evaluation (GPE) and generalized policy improvement (GPI), which are generalizations of two key operations of RL algorithms (policy evaluation and policy improvement), the agents are able to construct policies for any given preference over objectives. Therefore, this work proposes a transfer learning (TL) method for multi-objective multi-agent reinforcement learning (MOMARL). This helps fill a gap within the literature, since the vast majority of the RL literature focuses on single-agents with single-objectives, and only a handful of works address MOMARL settings.

The main contributions of this work can be summed up as follows:

- This is the first work (to the authors' best knowledge) that leverages generalized policy improvement for settings with multiple agents and objectives.
- We empirically evaluate our method in a multi-objective multi-agent traffic signal control environment with traffic controllers optimizing for vehicles' and pedestrians' objectives, which has rarely been addressed in the traffic signal control literature.

The reader can find a discussion on the main underlying concepts behind this work and a review of the related literature in Section 2 and in Section 3, respectively. In Sec-

tion 4, the proposed method is presented and explained in details. Section 5 presents the experimental settings and discusses the results. Finally, Section 6 concludes this work.

2. Theoretical Background

This section presents underlying concepts on RL, MARL, and MORL, as well as on specific concepts such as Successor Features (SFs), Generalized Policy Evaluation (GPE) and Generalized Policy Improvement (GPI).

2.1. Reinforcement Learning

An RL problem can be formulated as a Markov Decision Process (MDP), which is a mathematical model for sequential decision making. An MDP can be formally defined as a tuple of the form $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, where \mathcal{S} is a set of states, \mathcal{A} is a set of actions, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a state transition function mapping state transitions caused by actions to probabilities, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$ is a reward function, and $\gamma \in [0, 1)$ is the discount factor. At each time step t , an agent observes a state S_t , selects and takes an action A_t , receives a reward R_{t+1} and transitions to state S_{t+1} according to the state transition function. A policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ defines how the agent behaves in the environment by mapping states in \mathcal{S} to actions in \mathcal{A} .

As aforementioned, agents aim to maximize the reward received. However, since RL deals with sequential decision making, at time step t , the best decision may not necessarily be to select an action that will probably lead to the greatest next reward. As the real objective of an RL agent is to maximize the expected discounted sum of all rewards received by the agent, a discount factor γ is used to discount the impact of rewards received in later time steps, thus determining the present value of a future reward.

The state-value function of a state s under a policy π , which is denoted by $V^\pi(s)$, is the expected value of all returns received by an agent that starts in state s and follows the policy π . This function is given by Eq. 1.

$$V^\pi(s) = \mathbb{E}^\pi \left[\sum_{i=0}^{\infty} \gamma^i R_{t+i+1} \middle| S_t = s \right]. \quad (1)$$

Similarly, the action-value function for a policy π , which is the expected return of taking action a in state s and then following the policy π , is denoted by $Q^\pi(s, a)$ and can be defined by the Bellman equation in Eq. 2.

$$Q^\pi(s, a) = \mathbb{E}^\pi \left[R_{t+1} + \gamma Q^\pi(S_{t+1}, \pi(S_{t+1})) \middle| S_t = s, A_t = a \right]. \quad (2)$$

A policy π is better than or equal to a policy π' if and only if the state-value function of any state s under policy π is greater than or equal to the state-value function for the same state under policy π' . This relation, shown in Eq. 3, induces a complete ordering over policies.

$$\pi \succeq \pi' \iff \forall s, V^\pi(s) \geq V^{\pi'}(s). \quad (3)$$

If a policy is better than or equal to all others, it is an optimal policy, denoted by π^* . It is important to note that there is always at least one optimal policy. Therefore, solving

a problem in RL means finding an optimal policy (or at least a policy that sufficiently approximates an optimal policy).

All optimal policies share the same state-value function (which is the optimal state-value function, denoted V^*) and the same action-value function (which is the optimal action-value function, denoted Q^*). Importantly, if the optimal action-value function is known, an optimal policy π^* can be defined as in Eq. 4.

$$\pi^*(s) \in \arg \max_{a \in \mathcal{A}} Q^*(s, a). \quad (4)$$

An important subset of RL algorithms are based on Dynamic Programming (DP) [7]. These methods convert Bellman equations into update rules [33], and make use of mathematical properties of MDPs to decrease the complexity of searching for optimal policies. RL methods based on DP rely on two fundamental operations: policy evaluation and policy improvement. Policy evaluation is the computation of the value function of a policy π , and policy improvement refers to finding a policy π' that is better than π .

2.2. Multi-Agent Reinforcement Learning

In order to model multi-agent decision making problems, MDPs are extended to Multi-Agent Systems (MAS) as Stochastic Games (SG) [30], which can be formally defined as a tuple $\langle n, \mathcal{S}, \mathcal{A}_{1..n}, \mathcal{T}, \mathcal{R}_{1..n}, \gamma \rangle$, where n is the number of agents, \mathcal{S} is the state space, \mathcal{A}_i is the set of actions of agent i ($\mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_n$ is the joint action space), $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is a state transition function, \mathcal{R}_i is the reward function of agent i , and $\gamma \in [0, 1)$ is the discount factor.

Differently than single-agent RL, in MARL each reward received by an agent depends not only on its own actions, but on the actions of other agents. Thus, as previously mentioned, several convergence guarantees that are true in single-agent RL no longer hold.

Not only do other agent's actions influence the reward received by each agent, but in fact agents may have opposing objectives, and the action of one agent might negatively impact the rewards received by another. Generally, the nature of a multi-agent task is classified as being either cooperative, competitive or mixed [9]. In MARL, agents can be trained in a centralized manner using, for instance, a shared global reward. However, this approach does not scale due to the curse of dimensionality, as well as because agents have difficulty understanding and distinguishing their own contribution to the larger system [26]. Another option is full decentralization, that is, agents learn and act individually, in a fully decentralized manner. A third option, is a middle ground between centralized and fully decentralized, in which agents learn and act in a decentralized manner, but employ knowledge transfer strategies in order to accelerate learning [31]. In this work, we follow this approach by allowing agents to share their learned policies, which are further combined via generalized policy improvement (Section 4.2).

2.3. Multi-Objective Reinforcement Learning

One common approach to enable agents to deal with multiple objectives is to combine all of the objectives into a single scalar reward function. Unfortunately, this rather simple solution has many drawbacks, as for instance it reduces drastically the explainability of

the model (as it will be hard to distinguish which particular objectives prompted a specific behavior), it is unable to handle many different types of preferences that might be required by a specific problem, and will require that the agents be retrained if preferences among objectives change [17].

With these drawbacks in mind, it becomes clear why, in order to deal with multiple objectives, an explicitly multi-objective approach is preferable over a scalarized reward approach (which is equivalent to a single-objective approach). Hence, instead of using a scalar reward, a vector-valued reward function will be used.

To formally describe a multi-objective decision making problem, MDPs are extended to the multiple objective setting as a Multi-Objective Markov Decision Process (MOMDP). An MOMDP only differs from an MDP in the reward function, which becomes a vector-valued function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}^d$ that specifies the immediate reward for each of the d objectives considered.

Since rewards are now vector-valued, state-value functions and action-value functions are now vector-valued as well, that is, $\mathbf{V}^\pi \in \mathbb{R}^d$ and $\mathbf{Q}^\pi \in \mathbb{R}^d$. For problems with single objectives, a policy π is either better, equal or worse than policy π' , as value functions completely order all policies. This is not necessarily the case in multi-objective decision making problems, as value functions under policy π , when compared to value functions under policy π' , might have a greater value for some objectives, but a smaller value for others. Because there is no complete ordering over policies, in MOMDPs value functions only offer a partial ordering over the policy space. Hence, in this setting, there can exist several possibly optimal value vectors \mathbf{V} and \mathbf{Q} . In order to deal with this, a few sets of possibly optimal policies and value vectors need to be defined.

Before the sets aforementioned are defined, the utility function (or scalarization function) $u : \mathbb{R}^d \rightarrow \mathbb{R}$ needs to be introduced. This function, shown in Eq. 5, maps a multi-objective state-value vector under a policy π to a scalar value. This function commonly takes the form of a linear combination.

$$V_u^\pi(s) = u(\mathbf{V}^\pi(s)). \quad (5)$$

According to a MOMDP problem taxonomy proposed by [29], the three major factors that classify the nature of an MOMDP problem are: (i) whether the goal is to find one or multiple policies; (ii) whether the utility function is a linear function or, more generally, any monotonically increasing function; (iii) whether stochastic policies (instead of only deterministic policies) are allowed. For the sake of the present discussion, more definitions will be given regarding factor (ii).

In the more general case, where the utility function is simply monotonically increasing, a set of viable policies commonly used in the literature is the Pareto front $PF(\Pi)$. A policy π Pareto-dominates another policy π' if its value is equal or greater for every objective, and strictly greater for at least one objective, according to Eq. 6. The Pareto front set is defined in Eq. 7. It is important to note that the Pareto front set is not necessarily the undominated set, as the Pareto front may be larger than the undominated set, depending on the utility function.

$$\mathbf{V}^\pi \succ_P \mathbf{V}^{\pi'} \iff \forall i, V_i^\pi \geq V_i^{\pi'} \wedge \exists i, V_i^\pi > V_i^{\pi'}. \quad (6)$$

$$PF(\Pi) = \{\pi \in \Pi \mid (\nexists \pi' \in \Pi) [\mathbf{V}^{\pi'} \succ_P \mathbf{V}^\pi]\}. \quad (7)$$

A linear utility function, shown in Eq. 8, is the inner product of a value vector \mathbf{V}^π and a vector of weights \mathbf{w} which adheres to the simplex constraints (that is, $\forall i \ w_i \geq 0$ and $\sum_i w_i = 1$) [28].

$$V_{\mathbf{w}}^\pi = \mathbf{w} \cdot \mathbf{V}^\pi. \quad (8)$$

If the utility function is linear, the undominated set of policies is a convex hull $\text{CH}(\Pi)$, defined in Eq. 9. Since this convex hull is an undominated set, it may contain policies in excess. Thus, a coverage set for the convex hull can be defined. A set $\text{CCS}(\Pi)$ is a *convex coverage set* [17] if it is a subset of $\text{CH}(\Pi)$ and if for every \mathbf{w} it contains a policy whose linearly scalarized value is maximal, as defined in Eq. 10.

$$\text{CH}(\Pi) = \{\pi \in \Pi \mid \exists \mathbf{w}, \forall \pi' \in \Pi : \mathbf{w} \cdot \mathbf{V}^\pi \geq \mathbf{w} \cdot \mathbf{V}^{\pi'}\}. \quad (9)$$

$$\text{CCS}(\Pi) \subseteq \text{CH}(\Pi) \wedge \left(\forall \mathbf{w}, \exists \pi \in \text{CCS}(\Pi), \forall \pi' \in \Pi : \mathbf{w} \cdot \mathbf{V}^\pi \geq \mathbf{w} \cdot \mathbf{V}^{\pi'} \right). \quad (10)$$

All of the sets defined in this Section are of extreme importance to multi-objective problems, because MORL algorithms seek to compute the policies contained in these sets in order to solve multi-objective decision-making problems. In this paper, we consider the linear utility case, and thus we tackle the problem of constructing an approximate CCS. In this case, we are inherently unable to identify solutions that lie on the concave region of the Pareto front [35].

2.4. Successor Features and Generalized Policy Improvement

In [5], it is argued that many complex problems tackled by RL can be broken down into multiple tasks, encoded by different reward functions. A reward function $\mathcal{R}_{\mathbf{w}}$ for a task is defined as in Eq. 11, where $\phi(s, a, s') : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}^d$ is an arbitrary function representing the features of the environment and $\mathbf{w} \in \mathbb{R}^d$ is a vector of weights.

$$\mathcal{R}_{\mathbf{w}}(s, a, s') = \mathbf{w} \cdot \phi(s, a, s'). \quad (11)$$

With this reward definition, given a policy π , the action-value function shown in Eq. 2 can be rewritten as in Eq. 12, where $\psi^\pi(s, a)$ are successor features (SFs) [4]. It is important to note that SFs satisfy a Bellman equation, so they can be computed using conventional RL algorithms.

$$\begin{aligned} Q_{\mathbf{w}}^\pi(s, a) &= \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k \mathbf{w} \cdot \phi_{t+k} \mid S_t = s, A_t = a, \pi \right] \\ &= \mathbf{w} \cdot \mathbb{E} \left[\sum_{k=0}^{\infty} \gamma^k \phi_{t+k} \mid S_t = s, A_t = a, \pi \right] \\ &= \mathbf{w} \cdot \psi^\pi(s, a). \end{aligned} \quad (12)$$

The definitions given above, combined with a generalization of two fundamental operations of DP methods for RL (policy evaluation and policy improvement, mentioned in Section 2.1), provide a framework for knowledge transfer among tasks.

Suppose a RL agent, using an arbitrary RL method, has, through training, learned policies for k tasks (given by k different instances of \mathbf{w}). The agent thus knows a set of policies $\Pi = \{\pi_i\}_{i=1}^k$ and a set of corresponding SFs $\Psi = \{\psi^{\pi_i}\}_{i=1}^k$.

Using Eq. 12, it is possible to compute the value-function of a policy π on the k tasks by simply computing a dot-product between the SFs, ψ^π , and each weight vector w . This is called generalized policy evaluation (GPE). Also, given a new task, generalized policy improvement (GPI) refers to finding a policy that is better than all of the policies in \mathcal{I} . Note that GPE and GPI generalize policy evaluation and policy improvement to multiple policies and tasks, so if $k = 1$, GPE and GPI are equivalent to the standard DP operations.

Clearly, there is a strong similarity between the SF framework and MORL problems encoded by separate reward functions. In fact, in [3] it is shown that the transfer problem addressed by SFs is equivalent to the multi-objective optimization in MORL. That work demonstrates this by showing that it is possible to map any SF problem to a MORL problem by converting each dimension of ϕ into an objective. That is, an MOMDP is created with $\mathcal{R}(s, a, s') = \phi(s, a, s')$. Importantly, in this case the definitions of SFs and multi-objective action-value function become equivalent, that is, $Q^\pi = \psi^\pi$.

Since there exists this equivalence between SF problems and MORL problems, it is possible to apply GPI to MORL. Furthermore, [3] also shows that, if a convex coverage set is learned, performing GPI on this set enables an RL agent to learn an optimal policy for any weight vector w .

3. Related Work

This section presents a review of the related literature. Section 3.1 and Section 3.2 address relevant works that have tackled SFs and GPI in MORL and MARL settings, respectively, and Section 3.3 discusses applications of RL for traffic signal control.

3.1. SFs and GPI in MORL

SFs and GPI are presented in [4] as a generalization of the concept of successor representation [11] and policy improvement, respectively. This provides a framework for reusing knowledge across RL tasks, which is further elaborated in [5].

By formally demonstrating the equivalence between the SF framework and MORL, [3] is able to make use of GPI within the context of MORL, thus proposing a method that is able to combine previously learned policies in order to compute an optimal policy for any preference of objectives. However, this work only deals with single-agent scenarios.

Leveraging GPI within MORL is achieved by first using SFOLS, a SFs based extension of Optimistic Linear Support (OLS) [28] to assign different weights to the RL agent, so that it iteratively computes a CCS (Eq. 10). Then, once a CCS is learned, applying GPI to this set enables the agent to generate an optimal policy for any possible preference of objectives.

Most of the works discussed in this section use as one of their experimental settings the Four-Room domain, which is an environment made up of four rooms (separated by walls) in which a single agent moves around and collects different types of objects, which relate to different objectives. Because this environment is such a common benchmark in the literature, this work extends the Four-Room environment to multiple agents, and uses this domain as one of the experimental settings.

3.2. SFs and GPI in MARL

Besides the works discussed in Section 3.1, only a handful of works have mentioned SFs in the context of MORL, such as [17] (which briefly states that SFs are a subclass of multi-objective decision making problems), and [1] (which also briefly mentions that SFs and MORL are analogous). However, there have been a few works that have extended the concepts of SFs and GPI to multi-agent scenarios.

Decomposing a problem using a multi-agent solution generally introduces a source of non-stationarity. One approach that has been used to tackle this is the centralized training with decentralized execution (CTDE) framework, in which the agents learn their policies together, but execute them independently.

This CTDE method is used in the universal value exploration (UNeVEn) algorithm, presented in [16], which extends universal successor features (USFs) [8] to multi-agent universal successor features (MAUSFs), and combines this with GPI to improve the joint exploration of agents during training. Furthermore, CTDE is also used in [21], which presents an approach that makes use of SFs and GPI, and enables knowledge transfer among tasks in MARL settings.

Another relevant work that tackles SFs using CTDE is [19]. They use SFs to isolate each agent’s impact on the performance of the entire system, which allows the method to create individual utility functions for each agent tailored to stabilize their training.

It is important to note that, since the works addressing SFs and GPI in MARL settings do so using the CTDE framework, they all suffer from scalability issues, for obvious reasons. The more agents in the environment, the more difficult it is to train these agents in a centralized manner. The method proposed in this work has no such scalability issues, as both execution and training occur in a decentralized fashion.

3.3. RL for Traffic Signal Control

Traffic signal controllers seek to determine a split of green times among various phases at an intersection. A phase is defined as a group of non-conflicting movements (e.g., flow in two opposite traffic directions) that can have a green light at the same time without conflict. There are many different ways that traffic controllers can go about making decisions, and [27] provides an overview of several of them.

The most basic form of traffic control is based on fixed times, where the split of green times among the phases is computed using historical data on traffic flow, if available. However, this approach is unable to adapt to changes in traffic demand, which may lead to an increase in waiting times. To mitigate this issue, an adaptive approach, such as RL, can be used to determine the split of green times using some measure of performance (for instance, accumulated waiting time).

The RL literature for traffic signal control is very extensive and diverse, and a detailed overview of different techniques is beyond the scope of this work. Thus, the reader is directed to surveys such as [6, 38, 36, 25].

Besides reducing the waiting time of vehicles, traffic signal controllers can also have different objectives, such as reducing queue lengths [12] and reducing the environmental impact of traffic by minimizing fuel consumption [18]. However, it must be emphasized that the literature on MORL for traffic signal control is small, and there are very few MORL works for signal control that address pedestrians, such as [13] and [39]. This is another literature gap that this work helps to fill.

4. Policy Transfer in MOMARL using GPI

This section describes the proposed method in detail. However, before the method is explained, some preliminary conditions need to be set forth. First, all agents must be homogeneous (that is, have the same sensors, possible actions, and objectives). We also assume that their utility functions are represented as an inner product between the objectives and the weights (Eq. 8). Finally, each agent can have its own individual weights w .

During the training phase, the method uses a multi-agent extension of the SFOLS algorithm [3] to distributively solve scalarized versions of the multi-objective problem, by assigning different weights to each agent. Each agent computes a policy that is stored in a set of policies shared by all agents, until they have computed a convex coverage set. This process is explained in Section 4.1.

Once the agents have decentrally computed a CCS, the training phase is over. Then, during the execution phase, each agent is able to compute an effective policy for any possible instance of w , by applying GPI [3] to the shared set. This is explained in Section 4.2. A pseudocode for the algorithm is shown in Algorithm 1.

4.1. Decentrally computing a shared set of policies

During training, the agents seek to distributively compute a set of policies, Π , in order to approximate a CCS. To do so, each agent a iteratively receives a different weight vector, w_a , and learns a policy, π_a , specialized to the multi-objective task w_a . Every time an agent receives a weight vector, it solves the task given the scalar reward function $\mathcal{R}_w(s, a, s') = w \cdot \mathcal{R}(s, a, s')$ using a multi-objective Q-learning algorithm. The policies learned by the agents (and their corresponding SFs) are stored in a shared set of policies, Π . This process is repeated until Π corresponds to a CCS. The key, then, is to know which weights to assign to the agents, and when to stop (that is, how to know when Π is a CCS). This is determined by using the OLS algorithm, which is an extension of Cheng's linear support algorithm [10].

Let Π be a set of policies shared among all agents and P a priority queue of weight vectors. At each iteration of the algorithm, the weights in P with greater priority are popped off the queue and assigned to the agents. If there are more agents than weights in P , then randomly sampled weights are assigned to the remaining agents. After each iteration, new weights are added, until Π corresponds to a CCS. If, at the end of an iteration, P is empty, that means that Π is a CCS, and the algorithm ends.

The first step is to add to P all weights in the extremum of the weight simplex (that is, all weights that have one component equal to 1 and all others equal to zero), and assign to these weights an infinite priority to ensure that they are the first weights to be trained on. After this initialization of the priority queue, the algorithm's main loop begins. The weights in P are popped off according to their priorities and assigned to the agents, which then proceed to learn policies for the scalarized version of their multi-objective task (using their respective weights to scalarize the task).

Once each agent a finishes learning its respective policy π_a (and corresponding SF ψ^{π_a}), the algorithm evaluates the multi-objective value, V^{π_a} , of each learned policy.

¹ Then, each policy (and SF) is added to the shared set Π , if it corresponds to a still unknown value vector (lines 20–22 of Algorithm 1).

Lastly, the algorithm determines the *corner weights*, which, as per Theorem 1, are the instances of \mathbf{w} that can provide the maximal improvement to the set of policies. Intuitively, corner weights are the instances of \mathbf{w} whose optimal scalarized value functions are farthest from the current known best value vector for that \mathbf{w} .

Theorem 1. [10] *Let \mathcal{W} be the set of all possible reward vectors, let CCS be a convex coverage set, and let Π be a set of deterministic policies. The maximum value of*

$$\max_{\mathbf{w} \in \mathcal{W}, \pi \in \text{CCS}} \min_{\pi' \in \Pi} \mathbf{w} \cdot \mathbf{V}^\pi - \mathbf{w} \cdot \mathbf{V}^{\pi'} \quad (13)$$

is at one of the corner weights of

$$V_{\mathbf{w}}^{\text{CB}} = \max_{\pi \in \Pi} \mathbf{w} \cdot \mathbf{V}^\pi \quad (14)$$

Let $V_{\mathbf{w}}^{\text{CB}}$, given by Eq. 14, be the current best value function. To determine the *corner weights*, the algorithm exploits the fact that $V_{\mathbf{w}}^{\text{CB}}$ is a piecewise linear and convex (PWLC) [28, 10] function. Thus, the corner weights are the points in which $V_{\mathbf{w}}^{\text{CB}}$ changes slope.

Each corner weight \mathbf{w}_c is inserted into P with priority $\Delta(\mathbf{w}_c)$ (line 27), which is an optimistic estimate of the greatest possible improvement caused by learning a policy specialized to \mathbf{w}_c . This priority is given by Eq. 15, where $\bar{V}_{\mathbf{w}}^*$ is an optimistic upper-bound for the optimal value function $V_{\mathbf{w}}^*$.

$$\Delta(\mathbf{w}_c) = \bar{V}_{\mathbf{w}}^* - V_{\mathbf{w}}^{\text{CB}} \quad (15)$$

After inserting the corner weights in P , the algorithm's main loop (pop off weights with greatest priority from P , assign them to the agents, learn policies, add corner weights to P) is repeated, until P is empty, which indicates that there are no more corner weights, and therefore Π forms a CCS.

Let us propose an example to better illustrate the method. For the sake of simplicity, suppose a single agent (extending this example to multiple agents is trivial, as the only difference would be that more than one policy would be learned at each iteration) with three actions (A, B and C) and two objectives. Suppose there are 5 states: s_0, s_1, s_2, s_3 and s_4 . s_0 is the initial state and s_4 is the terminal state.

For this simple example, at each time step, the agent always transitions to the same states, regardless of the action selected. Choosing different actions in each state only changes the reward received. The MOMDP for this example is shown in Figure 1. A run of the OLS algorithm for the example is shown graphically in Figure 2.

Since this example has only two objectives, and the weights adhere to the simplex constraints, only one of the dimensions of \mathbf{w} is sufficient to express all possible instances of weights ($w_0 = 1 - w_1$).

First, the method inserts into P the weight vectors $[1, 0]$ and $[0, 1]$, the weights in the extrema of the weight simplex, with infinite priority. $\mathbf{w} = [1, 0]$ is popped off P , and the best policy for this weight is computed. This is shown in Figure 2a.

¹ Regular policy evaluation for single-objective RL can be used to determine the value of a policy for each of its objectives.

Algorithm 1: Multi-Agent SFOLS (MA-SFOLS)

```

1  $\Pi \leftarrow \{\}; \mathcal{V} \leftarrow \{\}; W \leftarrow \{\}; P \leftarrow \{\};$ 
2 foreach weight  $w_e$  in extremum of weight simplex do
3   | Insert  $(w_e, \infty)$  into  $P$ ;
4 end
5 while  $P$  is not empty do
6   | foreach agent  $a$  do
7     | if  $P$  is not empty then
8       |    $w_a \leftarrow$  pop weight with highest priority from  $P$ ;
9     | else
10      |    $w_a \leftarrow$  random weight;
11     | end
12     |   Insert  $w_a$  into  $W$ ;
13     |   Assign  $w_a$  to agent  $a$ ;
14   | end
15   | Wait until agents learn policies for their current weights;
16   | foreach agent  $a$  do
17     |    $\pi_a, \psi_a \leftarrow$  last policy and SF computed by agent  $a$ ;
18     |    $V^{\pi_a} \leftarrow$  value vector computed by agent  $a$ ;
19     |    $w_a \leftarrow$  current weight of agent  $a$ ;
20     |   if  $V^{\pi_a} \notin \mathcal{V}$  then
21       |     Insert  $V^{\pi_a}$  into  $\mathcal{V}$ ;
22       |     Insert  $(\pi_a, \psi_a)$  into  $\Pi$ ;
23       |     Remove obsolete corner weights from  $P$ ;
24       |      $W_c \leftarrow$  getCornerWeights( $V^{\pi_a}, w_a, \mathcal{V}$ );
25       |     foreach  $w \in W_c$  do
26         |        $\Delta(w) \leftarrow$  getImprovementEstimate( $w, \mathcal{V}, W$ );
27         |       Insert  $(w, \Delta(w))$  into  $P$ ;
28       |     end
29     |   end
30   | end
31 end
32 return  $\Pi, \Psi$ 

```

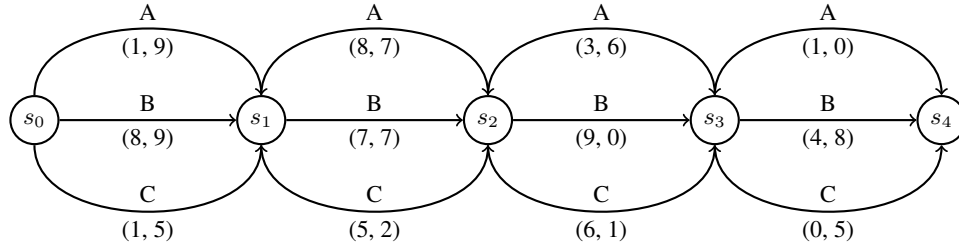


Fig. 1. Example: a simple MOMDP

Then, $w = [0, 1]$ is popped off P , and the best policy for this weight is computed. The corner weight $w = [0.32, 0.68]$ is found, with greatest possible improvement of $\Delta(w)$. This corner point is inserted in P , with a priority of $\Delta(w)$. This is shown in Figure 2b.

Next, $w = [0.32, 0.68]$ is popped off P , and the best policy for this weight is computed. The corner weight $w = [0.5, 0.5]$ is found, with greatest possible improvement of $\Delta(w)$. This corner point is inserted in P , with a priority of $\Delta(w)$. This is shown in Figure 2c.

Finally, $w = [0.5, 0.5]$ is popped off P , and the best policy for this weight is computed. Notice that the best policy for this weight is the same as the best policy for $w = [0.32, 0.68]$, so both curves overlap. No new corner point is found, and P is empty. The policies computed form a CCS, and the procedure ends. This is shown in Figure 2d.

4.2. Computing behaviors for new preferences using GPI

In Section 4.1, it is shown how the method makes use of a multi-agent extension of the OLS algorithm to decentrally compute a shared set of policies, and their respective successor features, which corresponds to a CCS. This section shows how the agents can create new policies for any possible weights using the policies in this shared set, thus transferring knowledge acquired from the source tasks (scalarized versions of the multi-objective problem using the weights in P) to the target tasks (scalarized versions of the multi-objective problem using new weights given to the agents during execution).

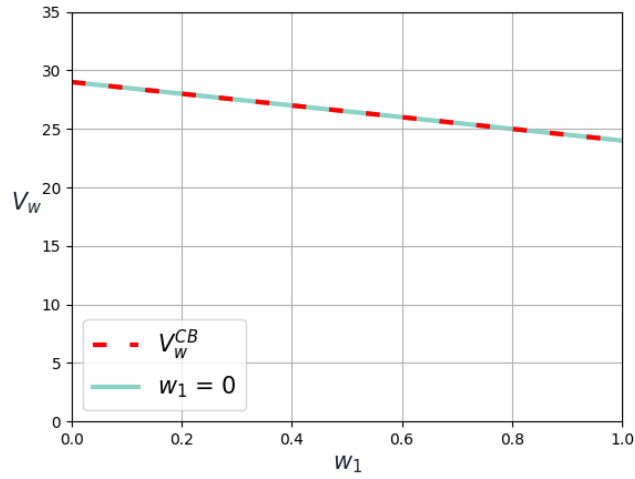
It is important to note that agents reuse knowledge acquired from policies learned by themselves and by others. This constitutes a type of multi-agent TL method, henceforth referred to as policy transfer (since previous policies are used to create new policies, both the source and target of the transfer are policies, thus policy transfer).

Policy improvement and GPI are only mentioned briefly in Section 2.1 and Section 2.4, respectively. Since they constitute a fundamental part of the proposed method, composing the core of the TL process, these operations are explained in more detail here, for the sake of a better order of presentation of information.

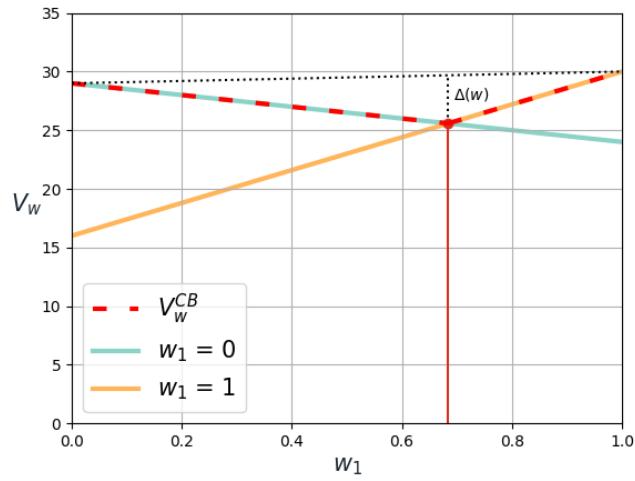
The *policy improvement theorem*, given by Theorem 2, is an extremely important result for RL. It shows that, for any state s , by selecting an action a in which $Q^\pi(s, a) > Q^\pi(s, \pi(s))$, a better policy emerges. This means that, by acting greedily with respect to the value function, policies can be improved upon.

Theorem 2. [33, 7] *Let π and π' be two deterministic policies such that, for any possible state s ,*

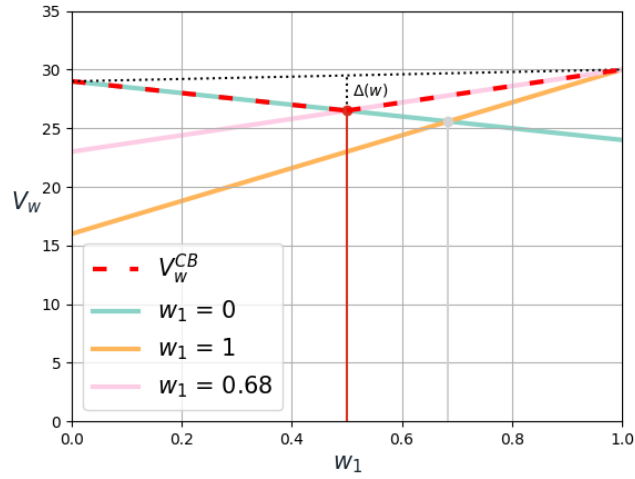
$$Q^\pi(s, \pi'(s)) \geq V^\pi(s). \quad (16)$$



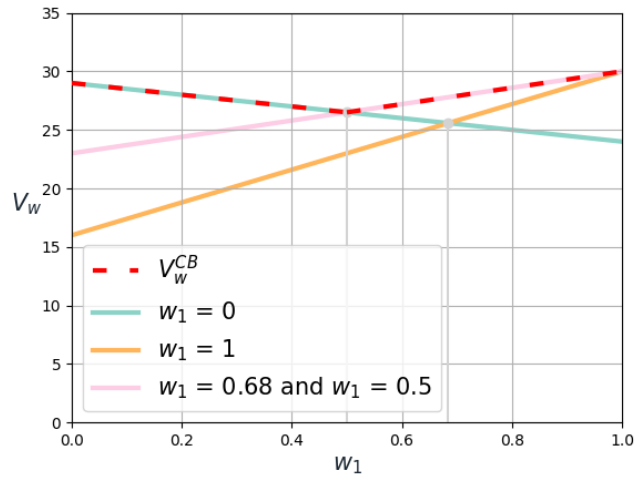
(a) Inserts weights in the extrema of the weight simplex with infinite priority. $w = [1, 0]$ is popped off



(b) $w = [0, 1]$ is popped off P . The corner weight $w = [0.32, 0.68]$ is found, and is inserted in P , with a priority of $\Delta(w)$



(c) $w = [0.32, 0.68]$ is popped off P . The corner weight $w = [0.5, 0.5]$ is found, and is inserted in P , with a priority of $\Delta(w)$



(d) $w = [0.5, 0.5]$ is popped off P . No new corner point is found, and P is empty. The policies computed form a CCS, and the procedure ends

Fig. 2. Computing a CCS for the example using OLS

Then, it holds that

$$\pi' \succeq \pi. \quad (17)$$

The operation of improving policies by acting greedily with respect to their value functions is called policy improvement, and is shown in Eq. 18.

$$\pi'(s) = \arg \max_{a \in \mathcal{A}} Q^\pi(s, a). \quad (18)$$

Within the framework of SFs, [4] extended Theorem 2 to a set of multiple policies. This is the *generalized policy improvement theorem*, given by Theorem 3.

Theorem 3. [4] *Let Π be a set of deterministic policies and let π' be a deterministic policy such that, for any possible state s ,*

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \quad \forall \pi \in \Pi. \quad (19)$$

Then, it holds that

$$\pi' \succ \pi \quad \forall \pi \in \Pi. \quad (20)$$

As aforementioned, the operation of improving a set of policies by acting greedily with respect to all of their value functions is called GPI, a generalization of policy improvement which was originally applied to the SFs framework. However, by showing the equivalence between SFs and MORL, [3] showed that it is possible to use GPI to improve policies in MORL settings. This result is key to the proposed method.

To understand how Theorem 3 can be applied to MORL, suppose the policies in Π correspond to policies learned by scalarizing a multi-objective problem using different weights. Now consider that a new weight vector is given to an agent. Because of the result in Theorem 3, this agent can create a new policy for this new weight vector by simply acting greedily with respect to the value function of all policies in Π for a scalarized version of the multi-objective problem using this new weight vector.

Now that a theoretical foundation has been laid, let us continue explaining the method. After having decentrally computed a shared set of policies Π , the agents are now ready to create new policies for any possible weight vector. Given a new weight \mathbf{w} , the agent uses Eq. 21 to generate a new policy, $\pi_{\mathbf{w}}^{\text{GPI}}$, best suited for the new weight vector, \mathbf{w} .

$$\pi_{\mathbf{w}}^{\text{GPI}}(s) \in \arg \max_{a \in \mathcal{A}} \max_{\pi \in \Pi} \mathbf{w} \cdot \psi^\pi(s, a). \quad (21)$$

Note that, as discussed in Section 2.4, when we define the features as being the multi-objective reward function ($\mathcal{R}(s, a, s') = \phi(s, a, s')$), then the SFs are equivalent to the multi-objective action-value function, that is, $\psi^\pi(s, a) = Q^\pi(s, a)$.

For a single-agent scenario, [3] proved that the policy $\pi_{\mathbf{w}}^{\text{GPI}}$ is optimal for \mathbf{w} if Π corresponds to a CCS. However, since we are interested in multi-agent scenarios, the convergence guarantees no longer hold. Nevertheless, Section 5 empirically shows that for complex MARL problems, even though $\pi_{\mathbf{w}}^{\text{GPI}}$ is not theoretically optimal, its performance is effective.

5. Experiments and Results

This section presents the experimental settings and results, and empirically shows that the method is both efficient at building a shared set of policies and effective at combining these policies using GPI to generate behaviors for different preferences over objectives. To evaluate the proposed method, two domains are used.

The Four-Room environment is used as one of the experimental settings in this work because it is employed in relevant related works, such as [4, 15, 3].

To show that the method also performs well for even more complex domains, and that the policies generated for new preferences over objectives are effective, a traffic signal control environment where traffic controllers optimize for both vehicles and pedestrians is also used, since this is an inherently distributed and multi-objective domain. Also, the agent’s actions in this environment are highly coupled, which makes it even more challenging.

As explained in Section 4.1, the agents can use any temporal difference learning algorithm to learn policies for the weights they receive. For our experiments, in both domains, multi-objective Q-learning with experience replay [14] was used. Table 1 shows the learning parameters used. Several values for each parameter were tested, but these were the ones that induced the best performance.

Table 1. Q-learning parameters.

Parameter	Value
α	0.1
ϵ	0.05
γ	0.95
Experience replay buffer size	1000000

5.1. Four-Room

The Four-Room environment was extended to contain multiple agents. Figure 3 depicts the environment. It is a 13×13 grid composed of four separate rooms, hence its name. The black squares represent the walls separating the rooms (the agents are unable to walk through these walls). There are three different types of objects (blue squares, green triangles and red circles), which are collected by the agents once they step on their corresponding squares. Each object corresponds to a different objective (there are, therefore, three different objectives). For this multi-agent extension, a second agent has been added. The agents start in the squares indicated by the labels S_1 and S_2 . Once they both reach the square indicated by the label G , the episode ends.

At each time step t , each agent observes a vector $s_t = [x, y]$ which represents the coordinates describing the agent’s current position in the grid. The agents are unaware of each other, but their actions are highly coupled.

Each agent has four possible actions: up, down, right and left (which move the agent one square in the corresponding direction). Agents are unable to step on black squares (which represent walls) and unable to leave the grid.

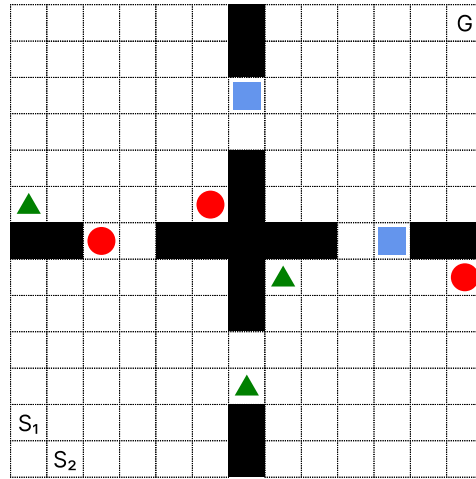


Fig. 3. Four-Room domain.

The reward $\mathcal{R}(s, a, s') \in \mathbb{R}^3$ is a three-dimensional vector representing the type of object in the agent's current cell ([1, 0, 0] if blue square, [0, 1, 0] if green triangle, [0, 0, 1] if red circle and [0, 0, 0] if blank). Once an agent steps on a square with an object, the agent receives the respective reward, and the object disappears, so the other agent can no longer receive a non-zero reward by stepping on that square. The agents, therefore, compete against each other.

This section aims at showing that the proposed method is efficient at generating a shared set of policies that can be effectively combined via GPI. Since the method iteratively assigns weights to the agents, which learn policies to scalarized versions of the multi-objective problem at each iteration, efficiency here can be measured by the number of iterations required to build the complete set of policies. A common baseline in the literature, which is used here as well, is to use random weights at each iteration [37].

Figure 4 shows the average return of both agents for 12 random test weights, after each iteration of MA-SFOLS + GPI and Random + GPI. As mentioned, for Random + GPI each agent is assigned a random weight at each iteration (instead of the corner weights, which offer the greatest possible improvement). For both MA-SFOLS and Random, GPI is used at every iteration to combine the policies learned so far to generate behaviors for the test weights.

From the plot, it is clear that the proposed method takes considerably less iterations in order to build an approximate CCS. For this example, the shared set of policies in most runs of the MA-SFOLS was already sufficient at iteration 4, whilst Random took more than twice as many iterations (approximately 10) to reach a similar level of returns.

The results empirically show the concept explained in Theorem 1: that corner weights offer the greatest possible improvement to expand the current set of policies. In fact, according to this theorem, no other weights assigned to the agents besides the ones selected by MA-SFOLS at each iteration could have built a CCS in fewer iterations.

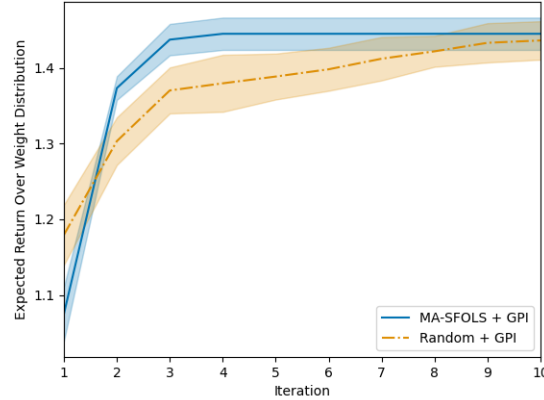


Fig. 4. Expected return of the agents over the distribution of reward weights (Four-Room).

5.2. Traffic Signal Control

The traffic scenario, shown in Figure 5, contains three intersections. Each traffic light at each intersection is controlled by an independent RL agent. The experiments were performed using SUMO-RL [2], which is based on the microscopic traffic simulator SUMO [22] (Simulation of Urban MOBility). SUMO-RL provides an interface for MARL, in which each agent can receive its own observations and rewards, and provide its own action to the traffic simulation. Below we detail how the scenario and the states, actions, and rewards of the agents were defined in our experimental setting.

The scenario selected is especially interesting and non-trivial because it considers pedestrians. Figure 6 shows a zoomed view of an intersection, in which lanes, vehicles, sidewalks, pedestrians, and a crossing can be seen. Notice that there is only one crossing lane per intersection, located at the road with greater vehicular demand. This is intentional, as it ensures that a conflict exists between the vehicle and the pedestrian objectives. Were there to be a crossing on the other road, the problem would become much less complex from a multi-objective perspective, as the same actions would benefit both objectives simultaneously.

Every link has 150 m in length, two lanes and is one-way. There are four Origin-Destination (OD) pairs: $VO_1 \rightarrow VD_1$, $HO_1 \rightarrow HD_1$, $HO_2 \rightarrow HD_2$ and $HO_3 \rightarrow HD_3$. Vehicles and pedestrians are inserted in an origin node and are removed from the simulation in a destination node. Vehicles travel in the North-South (N-S) direction in vertical links, and in the West-East (W-E) direction in horizontal links. Each vertical link has a pedestrian sidewalk, and each intersection has a crossing for pedestrians. Pedestrians only go in the North-South (N-S) direction.

Traffic signals in this scenario have a minimum and maximum green time. They are referred to as *minGreenTime* and *maxGreenTime*, respectively. For the experiments, these values were 10 and 50 seconds respectively. All signals have two phases, the North-South phase, which, when green, allows pedestrians and vehicles to flow in the North-South

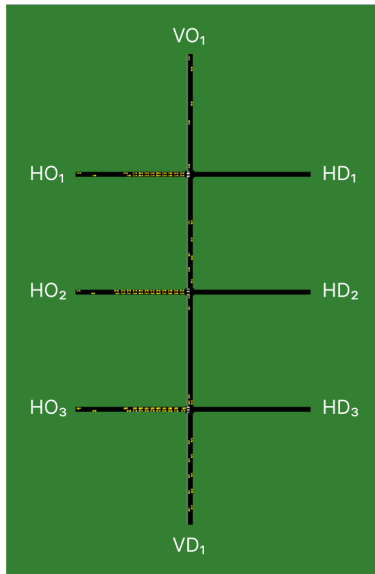


Fig. 5. Traffic signal control environment

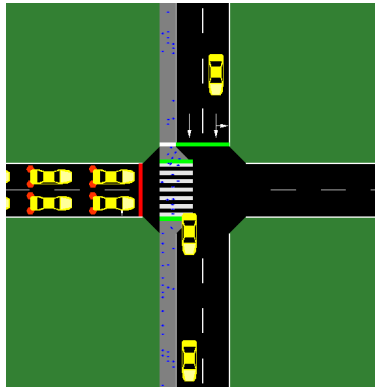


Fig. 6. Traffic intersection

direction, and the West-East phase, which, when green, allows vehicles to flow in the West-East direction.

Regarding demands, one vehicle is inserted in HO_1 , HO_2 and HO_3 every 3.3 seconds, and one vehicle is inserted in VO_1 every 20 seconds. As for pedestrians, one pedestrian is inserted every second in the origin node VO_1 . Each episode runs for 500 seconds. Once an episode ends, the simulation restarts.

At each time step t (which corresponds to five seconds of clock time), each agent observes a vector s_t , given by Eq. (22), which describes the current state of the respective intersection. $\rho \in \{0, 1\}$ is binary variable that indicates the current active green phase ($\rho = 0$ when the West-East phase is green, and $\rho = 1$ when the North-South and pedes-

trian phases are green). $\tau \in [0, 1]$ is the elapsed time of the current signal phase divided by $maxGreenTime$ (50 seconds). L is the set of all incoming lanes (for both vehicles and pedestrians). The density $\Delta_l \in [0, 1]$ is defined as the number of vehicles or pedestrians in the incoming lane $l \in L$ divided by the total capacity of the lane. $q_l \in [0, 1]$ is defined as the number of queued vehicles or pedestrians in the incoming lane $l \in L$ divided by the total capacity of the lane. A vehicle is considered to be queued if its speed is below 0.1 m/s. A pedestrian is considered to be queued if it is stopped before a crossing waiting for its phase to turn green.

$$s_t = [\rho, \tau, \Delta_1, \dots, \Delta_{|L|}, q_1, \dots, q_{|L|}] \quad (22)$$

Each signal controller agent chooses a discrete action a_t at each time step t . For our scenario, since all intersections have two incoming links, there are two phases, so each agent has only two actions: *keep* and *change*. The former keeps the current green signal active, while the latter switches the current green light to another phase. The agents can only choose *keep* if the current green phase has been active for less than $maxGreenTime$, and can only choose *change* if the current green phase has been active for at least $minGreenTime$.

For this scenario, the traffic controllers have two objectives: to minimize the waiting time of vehicles, and to minimize the waiting time of pedestrians. Thus, the reward function $\mathcal{R} \rightarrow \mathbb{R}^2$ is a two-dimensional vector, given by Eq. (23), where the first component is the change in cumulative vehicle waiting time between successive time steps, and the second component is the change in cumulative pedestrian waiting time between successive time steps.

$$\mathcal{R}_t = [Wv_t - Wv_{t+1}, Wp_t - Wp_{t+1}] \quad (23)$$

Wv_t is the cumulative vehicle waiting time at time step t , given by Eq. (24), where V_t is the set of incoming vehicles, and $w_{v,t}$ is the total waiting time of vehicle v since it entered the incoming road until time step t .

$$Wv_t = \sum_{v \in V_t} w_{v,t} \quad (24)$$

Wp_t is the cumulative pedestrian waiting time at time step t , given by Eq. (25), where P_t is the set of incoming pedestrians, and $w_{p,t}$ is the total waiting time of pedestrian p at the crossing until time step t .

$$Wp_t = \sum_{p \in P_t} w_{p,t} \quad (25)$$

Figure 7 shows the average return of all three traffic controllers for 25 random test weights,² after each iteration of MA-SFOLS + GPI. The performance of a few policies generated by GPI is shown in Figure 8, Figure 9 and Figure 10. Figure 8 shows the waiting time of pedestrians for the selected policies, and it is clear that the greater the value of w_1 (the weight corresponding to the pedestrian objective), the smaller the waiting time of pedestrians (thus, greater the performance for this objective). Figure 9 shows the waiting

² We uniformly sample weight vectors by sampling from a d -dimensional Dirichlet distribution ($\alpha = \mathbf{1}$), as in [1].

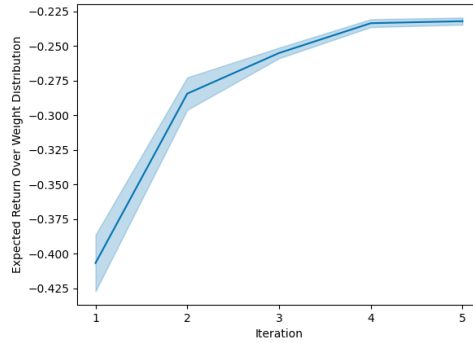


Fig. 7. Expected return of the agents over the distribution of reward weights

time of vehicles for the selected policies, and the same effect is visible (the greater the value of the weight corresponding to the vehicle objective, the better the performance for this objective). Figure 10 shows a zoomed view of Figure 9.

Note that the legends of the plots only show one dimension of the weight vector (since there are only two objectives, $w_0 = 1 - w_1$ and $w_1 = 1 - w_0$). If a plot shows waiting time of vehicles, the legend shows the weight dimension for vehicles (w_0), and if a plot shows waiting time of pedestrians, the legend shows the weight dimension for pedestrians (w_1).

It is clear from these plots how conflicting both objectives are (the higher the preference over one objective, the worse the performance is for the other objective). This makes this problem even more difficult, and also helps to serve as a motivation for multi-objective methods. Clearly, for this problem, a multi-objective method is required, as a priori scalarization of the problem would not be sufficient to address this scenario if the weights were not known in advance.

Section 5.1 showed the efficiency of the proposed method (measured by iterations required to form an approximate CCS). This section aims at showing the effectiveness of the method, that is, that the policies generated by GPI are effective for new reward weights. Thus, in the next experiments, we present an evaluation phase in which the set of policies identified during the training phase (via Algorithm 1) is shared with all agents. We compare the performance of policies generated via GPI with policies learned by Q-learning given a few selected reward weights.

Figures 11, 12, 13, 14 and 15 show a comparison between the performance (for both objectives) of policies generated by MA-SFOLS + GPI and the performance of policies learned by Q-learning via a priori scalarization for a few weights. It is clear that the policies generated by GPI have a very similar performance to the policies directly learned by Q-learning via a priori scalarization. Therefore, these results show that the method is able to appropriately leverage previous knowledge in order to create effective policies for new values of w .

The results in this section and in Section 5.1 empirically show that the method is both efficient and effective at building a shared set of policies that can be combined via GPI

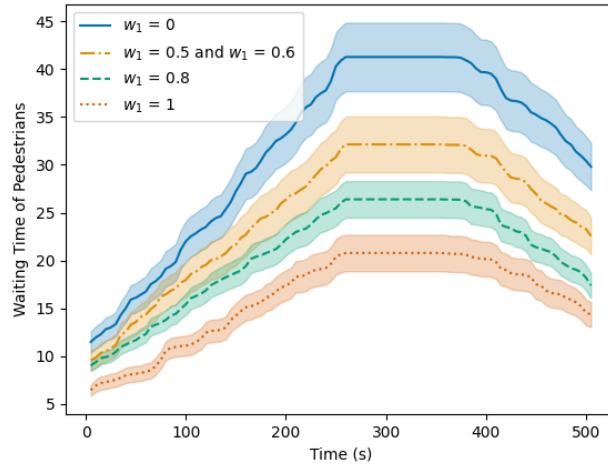


Fig. 8. Waiting time of pedestrians for a few policies generated by GPI

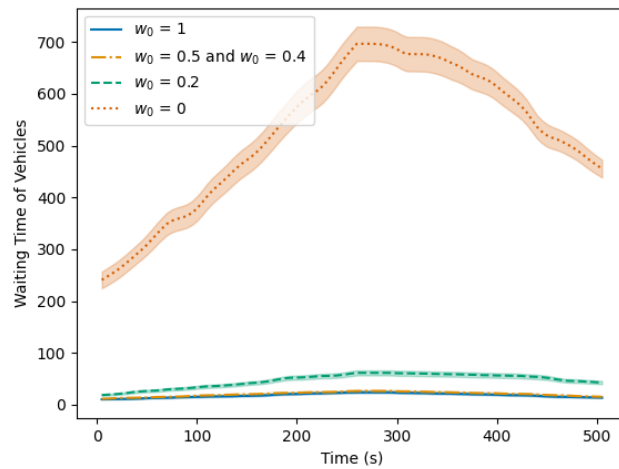


Fig. 9. Waiting time of vehicles for a few policies generated by GPI

to generate behaviors for new weights, so that the RL agents can perform well for any required preference of objectives.

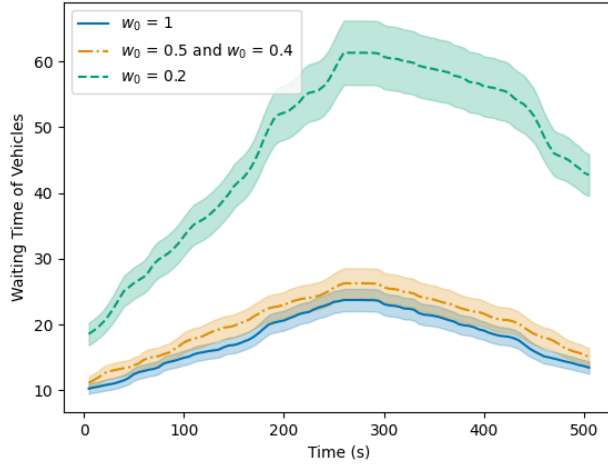


Fig. 10. Zoomed view of waiting time of vehicles

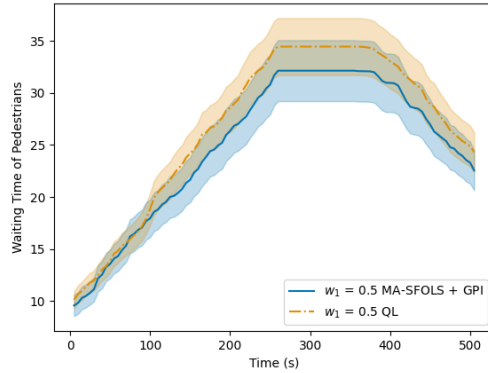


Fig. 11. MA-SFOLS + GPI vs Q-learning: $w = [0.5, 0.5]$ (pedestrians)

6. Conclusion

This work introduced a multi-objective multi-agent transfer learning method in which homogeneous agents decentrally build a shared set of policies during training. During the execution/evaluation phase, the agents combine these policies via GPI to create new policies specialized to any new linear combinations of their objectives. Within the method, two layers of knowledge transfer can be discerned: knowledge sharing and policy transfer.

Knowledge sharing refers to the fact that the policies learned by one agent can be used by another, thus all the agents in the system share their knowledge with each other. Policy transfer refers to the fact that, by leveraging a generalization of policy improvement, the

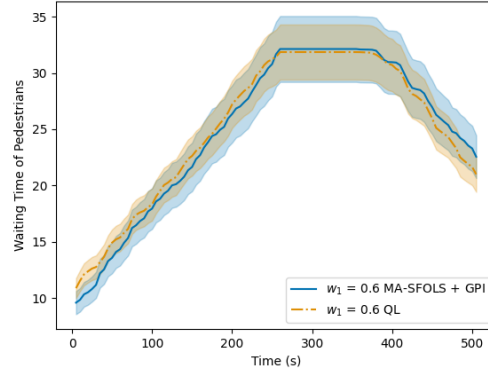


Fig. 12. MA-SFOLS + GPI vs Q-learning: $w = [0.4, 0.6]$ (pedestrians)

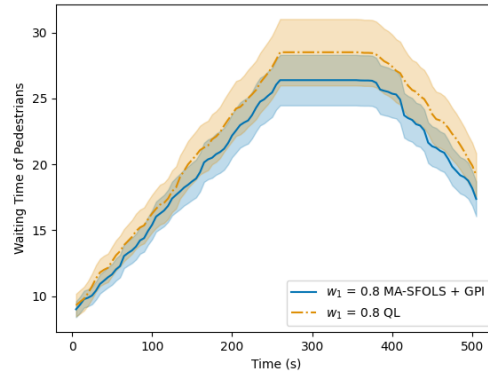


Fig. 13. MA-SFOLS + GPI vs Q-learning: $w = [0.2, 0.8]$ (pedestrians)

proposed method also enables agents to combine policies learned by themselves and by the other agents in order to create new policies, thus, policy transfer.

The proposed method is very useful for problems that are inherently distributed (therefore, a multi-agent solution is preferable or required) and multi-objective, and scalarizing them is not an option, either because the weights for the objectives are not known during training, or because the trade-off between objectives may change over time.

A traffic signal control domain with vehicles and pedestrians was used to evaluate the method, in addition to a standard domain in the SFs literature. It must be noted that optimizing for pedestrians is rarely addressed in the RL literature for traffic signal control.

The results empirically showed that the method is both efficient and effective at building a shared set of policies that can be combined via GPI to generate behaviors for new weights, so that the RL agents can perform well for any required preference of objectives.

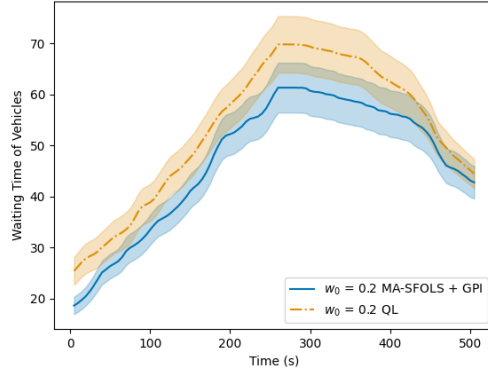


Fig. 14. MA-SFOLS + GPI vs Q-learning: $w = [0.2, 0.8]$ (vehicles)fig:four-room-returns

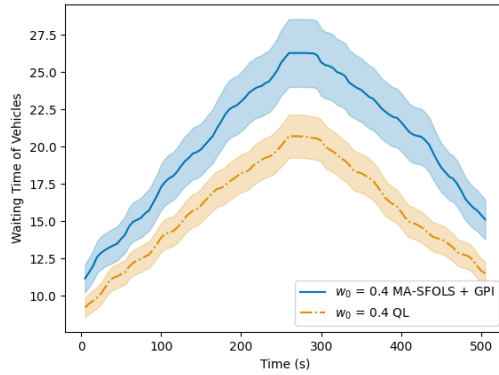


Fig. 15. MA-SFOLS + GPI vs Q-learning: $w = [0.4, 0.6]$ (vehicles)fig:four-room

This is one of the first works to address TL in the context of MOMARL, and the first work (to the authors’ best knowledge) that leverages generalized policy improvement for settings with multiple objectives and multiple agents.

In future work, we would like to address the two main limitations of our approach. First, we assumed linear utility functions, and thus our method is only able to identify policies in the CCS. How to deal with the more general case of non-linear utility functions is still an open problem in MORL [34]. Second, we considered independent learning agents that do not take into account the effects of non-stationarity caused by other agents. Extending our method in a way such that agents’ policies are conditioned on the preferences of other agents is a promising research direction. In order to tackle this problem, we plan to design a method that learns an inherently multi-agent coverage set, instead of a regular CCS; that is, a coverage set that takes into account the preferences over objectives of all agents simultaneously.

Acknowledgments. Ana Bazzan is partially supported by CNPq (grant 304932/2021-3). This work was partially funded by FAPESP and MCTI/CGI (grants number 2020/05165-1) and by CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil, Finance Code 001), and also partially sponsored by the German Federal Ministry of Education and Research (BMBF), Käte Hamburger Kolleg Cultures des Forschens/ Cultures of Research. We are grateful to the anonymous reviewers inputs.

References

1. Abels, A., Roijers, D.M., Lenaerts, T., Nowé, A., Steckelmacher, D.: Dynamic weights in multi-objective deep reinforcement learning. In: Proceedings of the 36th International Conference on Machine Learning. vol. 97, pp. 11–20. International Machine Learning Society (IMLS) (2019)
2. Alegre, L.N.: SUMO-RL. <https://github.com/LucasAlegre/sumo-rl> (2019)
3. Alegre, L.N., Bazzan, A.L.C., da Silva, B.C.: Optimistic linear support and successor features as a basis for optimal policy transfer. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 394–413. PMLR (17–23 Jul 2022), <https://proceedings.mlr.press/v162/alegre22a.html>
4. Barreto, A., Dabney, W., Munos, R., Hunt, J.J., Schaul, T., van Hasselt, H.P., Silver, D.: Successor features for transfer in reinforcement learning. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017)
5. Barreto, A., Hou, S., Borsa, D., Silver, D., Precup, D.: Fast reinforcement learning with generalized policy updates. Proceedings of the National Academy of Sciences 117(48), 30079–30087 (2020)
6. Bazzan, A.L.C.: Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. Autonomous Agents and Multiagent Systems 18(3), 342–375 (June 2009)
7. Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton (1957)
8. Borsa, D., Barreto, A., Quan, J., Mankowitz, D.J., Munos, R., Hasselt, H.V., Silver, D., Schaul, T.: Universal successor features approximators. In: Proceedings of the 7th International Conference on Learning Representations (ICLR) (2019)
9. Buşoniu, L., Babuska, R., De Schutter, B.: A comprehensive survey of multiagent reinforcement learning. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 38(2), 156–172 (2008)
10. Cheng, H.T.: Algorithms for partially observable Markov decision processes. Ph.D. thesis, University of British Columbia (1988), <https://open.library.ubc.ca/collections/ubctheses/831/items/1.0098252>
11. Dayan, P.: Improving generalization for temporal difference learning: The successor representation. Neural Computation 5(4), 613–624 (1993)
12. Duan, H., Li, Z., Zhang, Y.: Multiobjective reinforcement learning for traffic signal control using vehicular ad hoc network. EURASIP Journal on Advances in Signal Processing 2010 (12 2010)
13. Egea, A.C., Connaughton, C.: Assessment of reward functions in reinforcement learning for multi-modal urban traffic control under real-world limitations (2020), arXiv preprint arXiv:2010.08819
14. Fedus, W., Ramachandran, P., Agarwal, R., Bengio, Y., Larochelle, H., Rowland, M., Dabney, W.: Revisiting fundamentals of experience replay. In: Proceedings of the 37th International Conference on Machine Learning. Vienna, Austria (2020)
15. Gimelfarb, M., Barreto, A., Sanner, S., Lee, C.G.: Risk-aware transfer in reinforcement learning using successor features. In: Proceedings of the 35th Annual Conference on Advances in Neural Information Processing Systems. Online (2021)

16. Gupta, T., Mahajan, A., Peng, B., Böhmer, W., Whiteson, S.: Uneven: Universal value exploration for multi-agent reinforcement learning (2021), arXiv preprint arXiv:2010.02974
17. Hayes, C.F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L.M., Dazeley, R., Heintz, F., Howley, E., Irissappane, A.A., Mannion, P., Nowé, A., Ramos, G., Restelli, M., Vamplew, P., Roijers, D.M.: A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems* 36(1), 26 (Apr 2022), <https://doi.org/10.1007/s10458-022-09552-y>
18. Khamis, M.A., Gomaa, W.: Enhanced multiagent multi-objective reinforcement learning for urban traffic light control. In: 2012 11th International Conference on Machine Learning and Applications. vol. 1, pp. 586–591 (2012)
19. Kim, S.H., Stralen, N.V., Chowdhary, G., Tran, H.T.: Disentangling successor features for coordination in multi-agent reinforcement learning. In: International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022. pp. 751–760 (2022)
20. Kober, J., Bagnell, J.A., Peters, J.: Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32(11), 1238–1274 (2013)
21. Liu, W., Niu, D., Dong, L., Sun, C.: Efficient exploration for multi-agent reinforcement learning via transferable successor features. *IEEE/CAA Journal of Automatica Sinica* 9 (2022)
22. Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E.: Microscopic traffic simulation using SUMO. In: The 21st IEEE International Conference on Intelligent Transportation Systems (2018)
23. Mazyavkina, N., Sviridov, S., Ivanov, S., Burnaev, E.: Reinforcement learning for combinatorial optimization: A survey. *Computers and Operations Research* 134, 105400 (2021)
24. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. *Nature* 518(7540), 529–533 (Feb 2015)
25. Noaen, M., Naik, A., Goodman, L., Crebo, J., Abrar, T., Far, B., Abad, Z.S.H., Bazzan, A.L.C.: Reinforcement learning in urban network traffic signal control: A systematic literature review (2021), engrxiv.org/ewxrj
26. Rădulescu, R., Mannion, P., Roijers, D., Nowé, A.: Multi-objective multi-agent decision making: a utility-based analysis and survey. *Autonomous Agents and Multi-Agent Systems* 34 (04 2020)
27. Roess, R.P., Prassas, E.S., McShane, W.R.: *Traffic Engineering*. Prentice Hall, 3rd edn. (2004)
28. Roijers, D.: *Multi-Objective Decision-Theoretic Planning*. Ph.D. thesis, University of Amsterdam (2016)
29. Roijers, D.M., Vamplew, P., Whiteson, S., Dazeley, R.: A survey of multi-objective sequential decision-making. *J. Artificial Intelligence Research* 48(1), 67–113 (Oct 2013)
30. Shapley, L.S.: Stochastic games. *Proceedings of the National Academy of Sciences* 39(10), 1095–1100 (1953)
31. Silva, F.L.d., Costa, A.H.R.: A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research* 64, 645–703 (2019)
32. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., Hassabis, D.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm (2017)
33. Sutton, R.S., Barto, A.G.: *Reinforcement learning: An introduction*. The MIT Press, second edn. (2018)
34. Vamplew, P., Foale, C., Dazeley, R.: The impact of environmental stochasticity on value-based multiobjective reinforcement learning. *Neural Computing and Applications* (Mar 2021)
35. Vamplew, P., Yearwood, J., Dazeley, R., Berry, A.: On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts. In: Wobcke, W., Zhang, M. (eds.) *AI 2008: Advances in Artificial Intelligence*. pp. 372–378. Springer, Berlin, Heidelberg (2008)
36. Wei, H., Zheng, G., Gayah, V.V., Li, Z.: A survey on traffic signal control methods (2020), <http://arxiv.org/abs/1904.08117>, preprint arXiv:1904.08117

37. Yang, R., Sun, X., Narasimhan, K.: A generalized algorithm for multi-objective reinforcement learning and policy adaptation. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) *Advances in Neural Information Processing Systems* 32. pp. 14610–14621 (2019)
38. Yau, K.L.A., Qadir, J., Khoo, H.L., Ling, M.H., Komisarczuk, P.: A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Comput. Surv.* 50(3) (2017)
39. Yin, B., Menendez, M.: A reinforcement learning method for traffic signal control at an isolated intersection with pedestrian flows. pp. 3123–3135 (07 2019)
40. Yu, C., Liu, J., Nemati, S., Yin, G.: Reinforcement learning in healthcare: A survey. *ACM Comput. Surv.* 55(1) (nov 2021)

Vicente Almeida received the B.Sc. degree (cum laude) in computer engineering from the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil, in 2022, where he is currently pursuing the masters degree with the Institute of Informatics. His research interests include data exploration, reinforcement learning, and visual analytics. In his masters, he is currently working on the problem of exploring large datasets by combining multiple hypothesis testing with data-informed dimensions.

Lucas N. Alegre received the B.Sc. degree (cum laude) in computer science from the Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil, in 2021, where he is currently pursuing the Ph.D. degree with the Institute of Informatics. Part of his Ph.D. was done in the AI Lab at the Vrije Universiteit Brussel (VUB). His research interests include reinforcement learning, machine learning, artificial intelligence, and their applications to real-world problems. In particular, in his Ph.D., he is tackling the problem of how to design sample-efficient multi-task and multi-objective reinforcement learning algorithms capable of learning multiple behaviors that can be combined to solve novel problems.

Ana Bazzan is a professor of Computer Science at the Institute of Informatics at the Universidade Federal do Rio Grande do Sul (UFRGS), where she leads the Artificial Intelligence Group. Her research focuses on multiagent systems, in particular on agent-based modeling and simulation, and multiagent learning. Since 1996, she has collaborated with various researchers in the application of multiagent systems. In recent years, she has contributed to different topics regarding smart cities, focusing on transportation. In 2014, she was General Co-chair of AAMAS (the premier conference in the area of autonomous agents and multiagent systems).

Received: December 10, 2022; Accepted: September 02, 2023.

3D Convolutional Long Short-Term Encoder-Decoder Network for Moving Object Segmentation *

Anil Turker¹ and Ender Mete Eksioglu²

¹ ASELSAN Inc.

Ankara, Turkey

anilturker17@gmail.com

² Electronics and Communication Engineering Department,

Istanbul Technical University, Istanbul, Turkey

eksioglu@itu.edu.tr

Abstract. Moving object segmentation (MOS) is one of the important and well-studied computer vision tasks that is used in a variety of applications, such as video surveillance systems, human tracking, self-driving cars, and video compression. While traditional approaches to MOS rely on hand-crafted features or background modeling, deep learning methods using Convolution Neural Networks (CNNs) have been shown to be more effective in extracting features and achieving better accuracy. However, most deep learning-based methods for MOS offer scene-dependent solutions, leading to reduced performance when tested on previously unseen video content. Because spatial features are insufficient to represent the motion information, the spatial and temporal features should be used together to succeed in unseen videos. To address this issue, we propose the MOS-Net deep framework, an encoder-decoder network that combines spatial and temporal features using the flux tensor algorithm, 3D CNNs, and ConvLSTM in its different variants. MOS-Net 2.0 is an enhanced version of the base MOS-Net structure, where additional ConvLSTM modules are added to 3D CNNs for extracting long-term spatiotemporal features. In the final stage of the framework the output of the encoder-decoder network, the foreground probability map, is thresholded for producing a binary mask where moving objects are in the foreground and the rest forms the background. In addition, an ablation study has been conducted to evaluate different combinations as inputs to the proposed network, using the ChangeDetection2014 (CDnet2014) which includes challenging videos such as those with dynamic backgrounds, bad weather, and illumination changes. In most approaches, the training and test strategy are not announced, making it difficult to compare the algorithm results. In addition, the proposed method can be evaluated differently as video-optimized or video-agnostic. In video-optimized approaches, the training and test set is obtained randomly and separated from the overall dataset. The results of the proposed method are compared with competitive methods from the literature using the same evaluation strategy. It has been observed that the introduced MOS networks give highly competitive results on the CDnet2014 dataset. The source code for the simulations provided in this work is available online.

Keywords: Moving object segmentation, flux tensor, deep learning, spatiotemporal, change detection, foreground segmentation, background subtraction.

*A preliminary version of this work has been published as a conference paper in INISTA 2022.

1. Introduction

Moving object detection and segmentation is a vital task in many systems, such as video surveillance, human tracking, action recognition, self-driving cars, and video compression. It is particularly important in video surveillance systems and is also used as a sub-system in applications like video compression and self-driving cars. Moving object segmentation, also known as background subtraction, involves modeling the background and then comparing it to the current frame in order to classify each pixel as foreground or background. The accuracy of the algorithm depends heavily on the background modeling and maintenance process. A background model is created using the initial background and updated using upcoming frames. There are different strategies used for updating the process of background. The update rate is one of the significant hyper-parameters of the algorithm that affect the accuracy.

Moving object segmentation (MOS) is considered a semantic segmentation problem, each pixel is classified as foreground or background. The encoder-decoder network is popularly used in semantic segmentation problems. The encoder part extracts features with convolutional filters and pooling operations, the decoder part uses high-level features to produce a binary map. In order to increase the performance of the decoder network, there is a skip connection for transferring low-level features to decoder layers that gives better localization results. However, spatial features alone are not sufficient for moving object detection or segmentation, so it is necessary to consider both spatial and temporal features.

Our first proposed network, called Moving Object Segmentation (MOS) Net, is designed for binary segmentation. It is based on the U-Net [1] network, which has been successful in many segmentation tasks and is frequently used in the literature. In this paper, we preferred to use a hybrid algorithm that contains a flux tensor and 3D CNN. Flux tensor method [2] is an efficient way to extract motion information without using eigenvalue decomposition. Also, the network contains a 3D CNN for extracting the spatiotemporal features by using temporal depth in the kernel. Combining the output of flux tensor and motion entropy maps extracted by convolutional filters makes the network more robust for unseen videos. The flux tensor output, the feature map extracted by the 3D CNN network, and the current frame are the inputs of our U-net model, and networks produce a foreground probability map as a result of the sigmoid activation in the last layer of the network. MOS-Net in its basic form was proposed in preliminary work [3].

This paper represents an extended version of the conference paper [3]. In this paper, we introduce the novel MOS-Net 2.0 network, which is an enhanced version of the vanilla MOS-Net. In MOS-Net 2.0, ConvLSTM modules are appended to 3D CNNs with the aim of extracting long-term spatiotemporal features. MOS-Net 2.0 greatly benefits from the utilization of these long-term spatiotemporal features. Simulation results indicate improved performance for the novel MOS-Net 2.0, especially in scenes that include dynamic background objects such as the shaking of moving tree leaves, snow, rain, waves, etc.

The rest of this paper is organized as follows. Section 2 presents a literature review that contains traditional and deep learning-based methods for moving object segmentation. Section 3 and Section 4 describe the architecture of our proposed encoder-decoder network variants which are called as MOS-Net and MOS-Net 2.0, respectively. Section 5 conducts a quantitative and qualitative comparison of our networks and the competing

methods from the literature and also includes an ablation study. Section 6 provides the conclusions and a final discussion of the obtained results.

2. Related Work

Traditional methods for detecting a moving object are also called background subtraction. The reason for this naming is that background modeled by the algorithm is compared with the current frame, and then each pixel is classified as a result of this comparison. We can examine the background subtraction algorithm in 3 parts: background initialization, background modeling and maintenance, and foreground segmentation. In the background initialization part, an initial background is initialized using historical video frames. The initialized background is a reference for segmentation in currently incoming video frames. An initial model can be created using the statistical properties of N video frames taken from the beginning of the video. Background modeling and its maintenance are the most critical part of the performance of the background subtraction algorithm. A background model is created using the initial background and updated using upcoming frames. Foreground segmentation is performed by comparing the background model and the current frame. There exist different comparison strategies. For instance, the absolute value of the difference between the current frame and the background model can be compared with a threshold value. If this difference is higher than the threshold value, the pixel is classified as foreground, otherwise as background.

There are many efficient and influential studies on moving object detection and segmentation in the literature. One important method is the Gaussian Mixture Model (GMM) [4], a parametric approach proposed by C. Stauffer et al. to model complex backgrounds such as moving background and brightness change. GMM is the weighted sum of N Gaussian distributions. In this method, the relationship between neighboring pixels is disregarded, and each pixel is evaluated independently. The mean value of the weighted Gaussian distribution is computed for each pixel and compared with the corresponding pixel value in the incoming frame. If a match is found, the pixel is classified as part of the background. GMM can tolerate the brightness changes because of the nature of Gaussian distribution and permit the different appearance of background objects with another Gaussian distribution for the pixel. ViBe [5], introduced by Barnich et al., is a method that models the background with a sample-based system. The last N values are kept in the library for each pixel. The classification process was carried out by comparing the pixels in this created library with newly arriving pixels. At least K matches are expected to classify a new pixel as background. The difference between the gray level values of the pixels has to be lower than a specified threshold value R to allow a match. Oliver et al [6] developed a learning-based algorithm in which they use subspace learning for modeling background, a method called as eigenbackground. The model first calculates the mean background image and covariance matrix from N sample images. Then the background is modeled using Principal Component Analysis (PCA) with the largest eigenvalues and corresponding M eigenvectors. After the new frames are projected on the eigenvectors, the picture and the projection are compared. Then the pixels are classified with a threshold value used in this comparison.

CNNs have been used by researchers to detect or segment moving objects, in addition to their extensive use in other fields of computer vision. Braham and Broeck proposed an

early method that uses convolutional filters for the background subtraction [7]. In their work, the background and current images are obtained by using the temporal median filter, and then they are fed into LeNet-5 [8] network in the form of $N*N$ patches. This method is significant for inspiring other researchers about the utilization of deep learning methods in this research area. Another popular method is DeepBs [9] which combines traditional and deep learning-based approaches and makes them more robust on unseen videos. It uses the Subsense algorithm [10] as a traditional approach. Then, the current frame and modeled background are given as input to the network in the form of $N*N$ patches. There is also a pixel library that contains the historical values of each pixel. The length of the library changes depends on the motion information generated by the flux tensor algorithm [2]. In contrast to DeepBs, Gao and Cai proposed an end-to-end network [11] that uses 3D convolution to extract both spatial and temporal features. FgSegNet [12] network has state-of-the-art results on the CDNet-2014 dataset. It is a segmentation network consisting of an encoder and decoder network that extracts multi-scale features. It uses 50 or 200 frames for each video in training the network. This algorithm provides a video-specific result for foreground segmentation.

Another recent work is the BSUV-Net network [13], which consists of a fully convolutional network and is tested on unseen videos. BSUV-Net is a network that has a similar network structure to U-net [1]. The network uses the current frame and two temporal median filters with different numbers for the historical frames. In addition to the three channels, they exploit the segmentation network in their proposed method. In most of the approaches, the training and test strategy are not announced, which makes it difficult to compare the algorithm results. In addition, the proposed method can be evaluated differently as video-optimized or video-agnostic. In video-optimized approaches, the training and test set is obtained from randomly separated video frames. Even if the datasets are selected from different video frames, they may contain similar images. Therefore, the network's performance on unseen videos is not sufficient. The training and test videos are entirely different in the video-agnostic methods, so the network test on unseen videos. Consequently, video-optimized approaches have an unfair advantage over video-agnostic techniques.

3. Proposed Method I - MOS-Net

This section gives details about our proposed network MOS-Net [3] consisting of 3D CNN, the flux tensor algorithm, and the encoder-decoder network. A preliminary version of this work including the MOS-Net was provided in [3]. The proposed network is shown in Figure 1. In addition, we give the details of our training strategy and loss function.

3.1. 3D CNNs

Motion features are extracted through 3D CNNs using consecutive frames. The last N frames go into the network, the temporal dimension is reduced in each convolution layer, and the motion information is extracted as a result of the network. The temporal stride parameter is used as 5, 5, and 2 in each convolution layer, respectively. The different kernel sizes 5×5 , 3×3 , and 1×1 are used for extracting multi-scale features, and these features maps are added up. We used 50 last recent frames as input frames in our proposed 3D CNNs, and its architecture is shown in Figure 1a.

3.2. The flux tensor algorithm

There are different algorithms used to obtain motion information in the literature. One of the most well-known of these is the Lucas Kanade method [14] for the estimation of optical flow. This algorithm makes the optical flow estimation by assuming that the motion in neighboring pixels is similar. The least-square fit method is used for estimating the motion vectors.

The flux tensor algorithm [2] is an extended version of the 3D grayscale structure tensor [15] which has coherent motion segmentation results with respect to classical optical flow methods. In the flux tensor algorithm [2], motion information can be extracted with a lower computational cost compared to other optical flow algorithms without using the eigenvalue decomposition. The local 3D spatiotemporal volume contains the optical flow field, and the flux tensor describes the changes in the field over time. The matrix of flux tensor is shown in Eq. (1). It is possible to distinguish between static and moving objects by using the flux tensor matrix. The trace of the matrix can be used directly to find the moving object in the current frame. The trace of the matrix is shown in Eq. (2)

$$\mathbf{J} = \begin{bmatrix} \int_{\Omega} \left(\frac{d^2 I}{dxdt}\right)^2 dy & \int_{\Omega} \frac{d^2 I}{dxdt} \frac{d^2 I}{dydt} dy & \int_{\Omega} \frac{d^2 I}{dxdt} \frac{d^2 I}{dt^2} dy \\ \int_{\Omega} \frac{d^2 I}{dydt} \frac{d^2 I}{dxdt} dy & \int_{\Omega} \left(\frac{d^2 I}{dydt}\right)^2 dy & \int_{\Omega} \frac{d^2 I}{dydt} \frac{d^2 I}{dt^2} dy \\ \int_{\Omega} \frac{d^2 I}{dt^2} \frac{d^2 I}{dxdt} dy & \int_{\Omega} \frac{d^2 I}{dt^2} \frac{d^2 I}{dydt} dy & \int_{\Omega} \left(\frac{d^2 I}{dt^2}\right)^2 dy \end{bmatrix} \quad (1)$$

$$\text{Trace}_J = \int_{\Omega} \left\| \frac{d}{dt} (\nabla I) \right\|^2 dy \quad (2)$$

3.3. Encoder-Decoder network

This study uses a U-Net type encoder-decoder network for foreground segmentation. BSUV-Net [13], one of the recent methods, uses the U-net type neural network for the segmentation process. It also utilizes reference frames obtained from the temporal median filter at different intervals. The current frame and reference frames are given as input to a U-net network. In our proposed network, unlike BSUV-Net, the motion information is extracted by using 3D CNNs and flux tensor algorithm. The extracted motion feature maps are given as input to the encoder-decoder network, together with the current frame. Our proposed encoder-decoder network for MOS-Net is shown in Figure 1b.

The encoder module reduces the spatial resolution by using convolution and max pooling operations. The decoder module upsamples the reduced form to the original resolution. The decoder part maps the low-resolution image containing the features to the original resolution and utilizes the skip connection in the network. The first four convolutional layers of the encoder module are the same as the VGG-16 [16] network. All convolution filters have a 3×3 kernel size and max-pooling has a 2×2 kernel size as a stride rate of 2. Batch normalization [17] is used at the end of each convolution layer and standardizes the input for each mini-batch. It stabilizes the network, speeds up the training process, and has a regularization effect.

Motion information obtained by using a flux tensor and 3D convolutional neural network is input to the encoder-decoder network together with the current frame. The network gives an output of a binary image with the same spatial resolution as the current

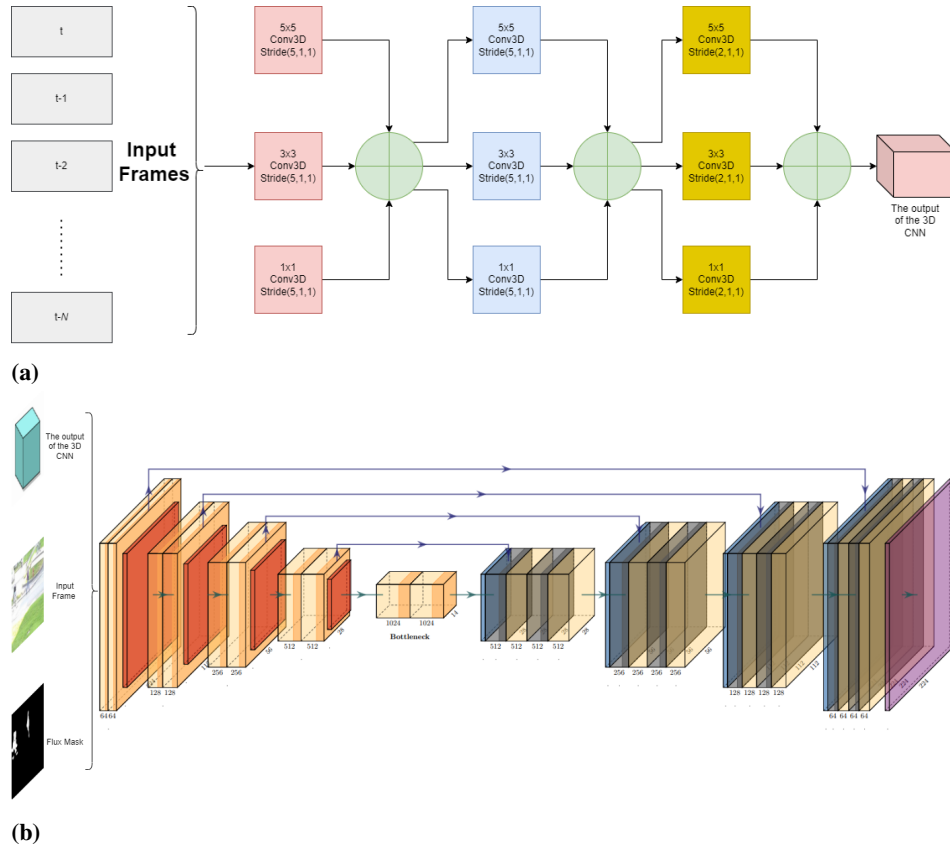


Fig. 1. Proposed foreground segmentation network, MOS-Net. (a) Initial 3D Convolutional Neural Network extracting the motion entropy maps from N historical frames, (b) Encoder-decoder network creating the final segmentation map output

frame. The binary image that contains the foreground probability map is obtained by using the sigmoid activation function. The foreground probability map consists of pixels value between 0 and 1, and pixels are threshold by a value; ones higher than the threshold value are classified as foreground, and others are the background.

3.4. Loss function

Moving object segmentation is fundamentally the classification of each pixel as foreground or background. The number of background pixels on the dataset is much more than the number of foreground pixels. This imbalance problem degrades the performance of the proposed fully convolutional network. The cross-entropy loss function is prevalent in semantic segmentation tasks. When this loss function is used in an imbalanced dataset, the classifier favors the majority classes, and the network will be a biased model. The weighted cross-entropy loss is proposed as a solution to this imbalance problem. The effect of the samples with a minority in the dataset on the loss function gets increased in the weighted cross-entropy loss. The Jaccard index or IoU (Intersection over Union) is used widely for imbalanced dataset problems in object detection tasks. In addition to object detection, this loss function can also be used for semantic segmentation tasks. The Jaccard index is computed as follows.

$$Jaccard\ Index = \frac{TP}{TP + FP + FN} \quad (3)$$

Here, the below given definitions are being used.

TP: the count of true positives
FP: the count of false positives
FN: the count of false negatives

The Jaccard index measures the sensitivity of foreground pixel segmentation results. As the performance of the segmentation network improves, this value approaches one. On the other hand, as the performance decreases, the index approaches zero. The Jaccard index can be utilized to define a loss function with the following equation.

$$Jaccard\ Loss\ Function = (1 - Jaccard\ Index) * \alpha \quad (4)$$

Here, α is a smoothing parameter.

4. Proposed Method II - MOS-Net 2.0

In the MOS-Net network introduced in Section 3, short-time spatiotemporal features are extracted from consecutive frames via a 3D Convolutional Neural Network. As stated in Chapter 2, the ConvLSTM time series can be used to extract long-term spatiotemporal features from an input. In order to increase the performance of the 3D CNN network used in this study, we add ConvLSTM modules to the 3D CNN spatiotemporal feature extractor. This lead to the improved MOS-Net 2.0 network. The full network structure for MOS-Net 2.0 is shown in Figure 2. The encoder-decoder network responsible for

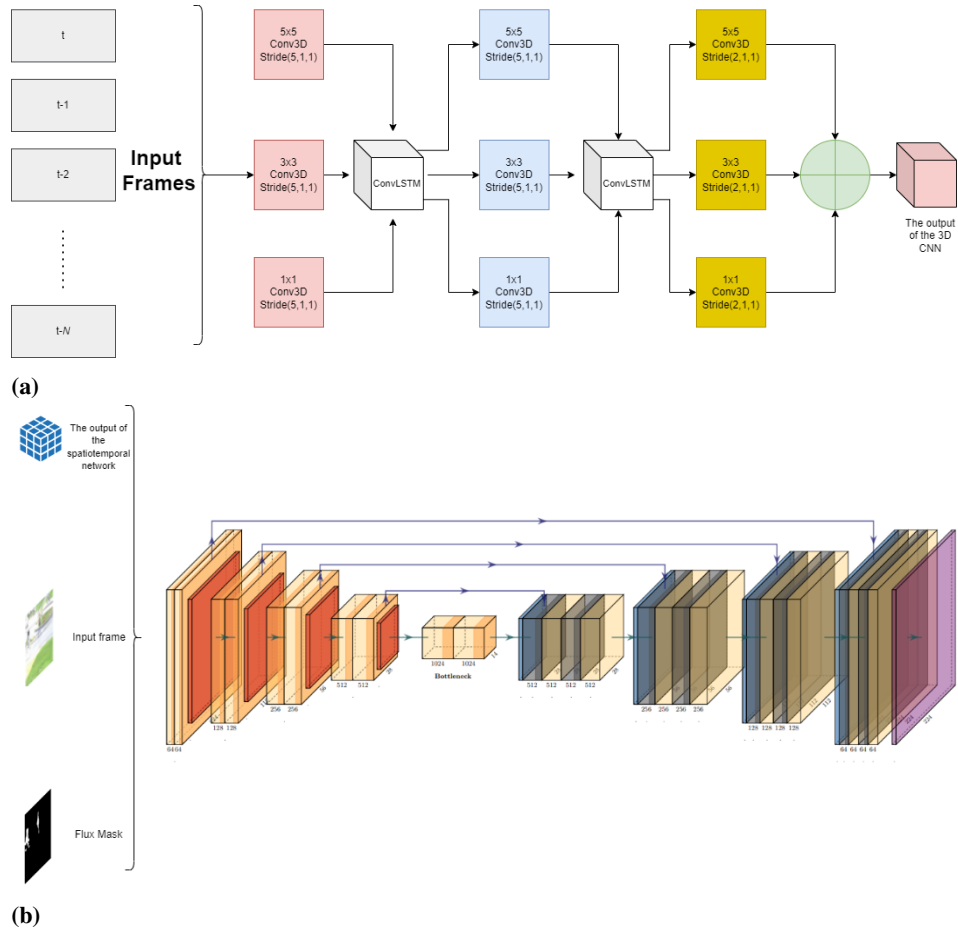


Fig. 2. Proposed foreground segmentation network, MOS-Net 2.0. a) Initial 3D CNN network with novel ConvLSTM modules generating the spatiotemporal features, b) Encoder-decoder network creating the final segmentation map

feature extraction and dense prediction, the flux tensor algorithm, and the utilized loss function are the same as in MOS-Net.

LSTM is one of the RNN methods used in time series problems. However, considering the large number of pixels in an image, too many parameters are required for an LSTM realization when imported to an image processing setting. At the same time, LSTM is insufficient to extract spatial features. Therefore, ConvLSTM is preferred where the matrix multiplication is exchanged with a convolution operation using a 2D kernel. ConvLSTM can extract spatiotemporal features from the 2D time series image, using both the convolutional layer and also the cell state included in LSTM. 3D CNNs transmit the necessary information for motion information to the encoder-decoder network. In the MOS-Net 2.0, the ConvLSTM module is applied to an element-wise collection of feature maps obtained with different sized filter in 3D convolutions. The updated architecture for the spatiotemporal feature extractor is shown in Figure 2a. The use of ConvLSTM is intended to improve the robustness of the network in extracting motion information.

5. Experiments and Performance Evaluation

The proposed method is trained and tested using the ChangeDetection2014 (CDnet2014) dataset [18], which is frequently used in moving object segmentation tasks. ChangeDetection is often employed in assessing the performance of moving object segmentation and change detection algorithms. CDnet2014 was announced as an extended version of the 2012 CDnet dataset, with five new categories and 22 videos (Figure 3).

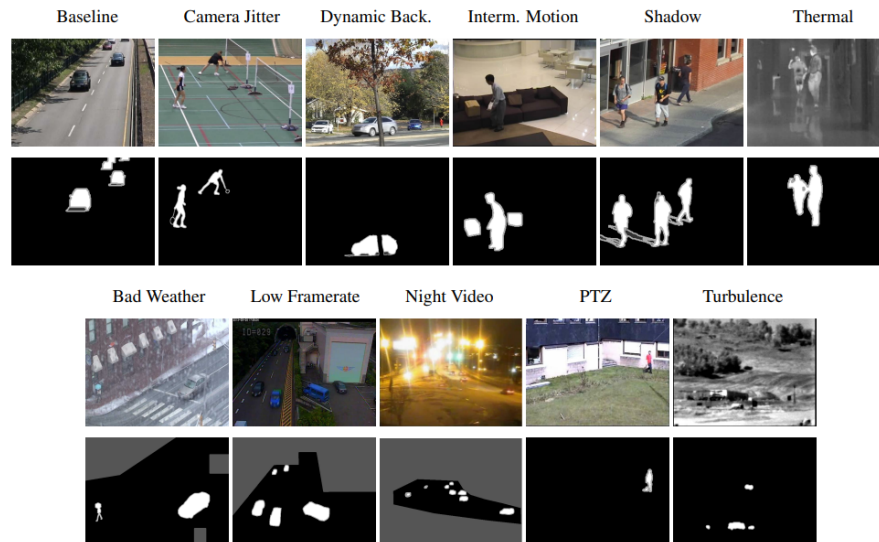


Fig. 3. The sample video frames from CDnet2014 dataset [18]

We implemented proposed methods using the Pytorch framework and a Tesla P100-PCI-E-16GB GPU, utilizing a batch size of 8. Adam was used as the optimization algorithm during the training, and the learning rate started at 0.0001. Afterwards the learning rate was gradually reduced every 20 epochs. The maximum number of epochs is 60 for each network configuration. The networks are trained by randomly selecting 200 video frames for each video listed in Table 1.

Fully convolutional networks are independent of the particular spatial resolution of the input. The spatial resolutions of the videos in the CD2014 dataset differ from each other. We used the fixed size input as 224×224 to utilize the parallel processing power of GPUs. The inputs can be made same size by resizing operations or cropping fixed-sized patches from the images. We used the randomly cropping strategy for the fixed-size input in the training process, since randomly cropped images give the network an extra augmentation and regularization effect. In addition, random noise has been added to the input image as another augmentation technique that makes the network robust to sudden changes. On the other hand, we used the original spatial resolution of the input frame without any scaling or cropping operation in the inference process.

Table 1. Change Detection 2014 Dataset Data Division [19]

Category	Train Data	Test Data
Baseline	highway, office, PETS2006	pedestrians
badWeather	skating, snowFall, wetSnow	blizzard
cameraJitter	badminton, boulevard, sidewalk	traffic
dynamicBackground	canoe, fall, fountain01, fountain02, overpass	boats
intermittentObjectMotion	abandonedBox, sofa, streetLight, tramstop, winterDriveway	parking
lowFramerate	port_0_17fps, tramCrossroad_1fps, tunnelExit_0_35fps	turnpike_00_05fps
nightVideos	bridgeEntry, busyBoulevard, fluidHighway, streetCornerAtNight, winterStreet	tramStation
shadow	backdoor, bungalows, cubicle, peopleInShade	busStation
thermal	diningRoom, lakeSide, library, park	corridor
turbulence	turbulence1, turbulence2, turbulence3	turbulence0

As a result of the encoder-decoder network, a foreground probability map (FPM) with values between 0 and 1 is obtained. Binary images can be produced using different threshold values such as 0.1, 0.3, 0.5, 0.7, and 0.9. Figure 4 shows the average F1 score of MOS-Net and MOS-Net 2.0 networks for these threshold values on the unseen videos in Table 1. It indicates that MOS-Net and MOS-Net 2.0 reach the best average F1 score with a 0.1 threshold value. Using these results, the fixed threshold value was determined as 0.1, and this threshold value was used in the experiments.

The network gives a foreground probability map resulting from the last sigmoid activation function. Then, pixel values greater than 0.1 classify as foreground and others as background. Our proposed methods, MOS-Net and MOS-Net 2.0 outperform competing methods as listed in Table 2. They have F1-scores 0.83 and 0.85 on the CDnet2014 dataset, respectively. As shown in Table 2, FgSegNet [12], a state of the art method for CDnet2014 dataset, has a performance of 0.22 in terms of F1 score. It has degraded performance on unseen videos, because FgSegNet [12] offers a video-specific solution. Another recent work is BSUV-Net [13]. Its performance is 0.80 in terms of F1-score on the unseen videos, as listed in Table 1. BSUV-Net [13] utilizes temporal median filters, which

suppresses the performance of the network for dynamic scenes. As can be seen from the results, our proposed network MOS-Net 2.0 gives the best results among the competing algorithms. In Table 3, visual results obtained in different categories of the CDnet2014 dataset are given.

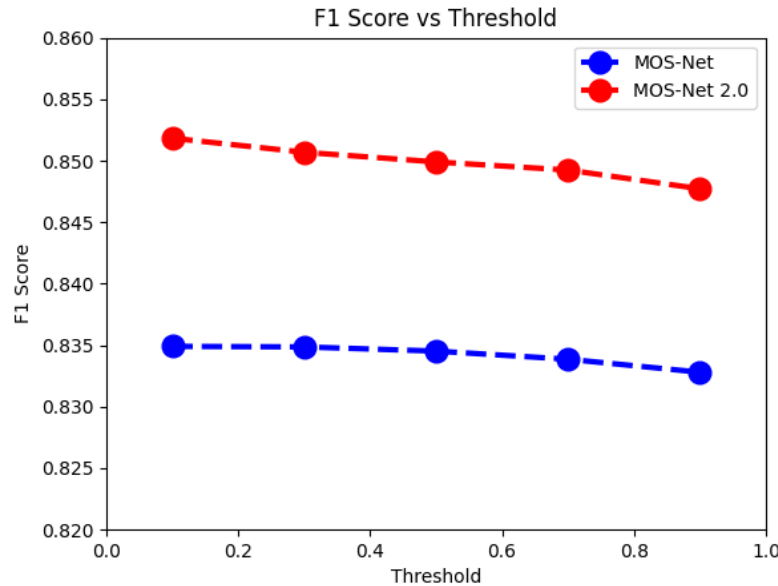


Fig. 4. The illustration of average F1 score on unseen videos versus thresholds

The comparison of the network parameter numbers of the proposed methods with FgSegNet and BSUV-Net, which are studies conducted in recent years, are given in Table 4. In the FgSegNet method, the parameters of the first three layers in the encoder block are frozen in the training process. These weights are the original weights of the VGG16 network. It is seen that FgSegnet has the lowest number of parameters since it uses convolutional filters and pooling operations for extracting only spatial features on the current frame. It has been observed that the performance on the unseen dataset is low due to the absence of spatiotemporal features. As seen in Table 4, the MOS-Net 2.0 with the highest F1 score also has the highest number of parameters. MOS-Net 2.0 has more parameters than MOS-Net, because of the inclusion of the ConvLSTM module.

One important contribution of this work is the evaluation of the effects of traditional and deep learning based features on the moving object segmentation performance. Our proposed networks, MOS-Net and MOS-NET 2.0 have multiple inputs at the encoder module, which extracts further spatiotemporal features using these inputs. We retrained the encoder-decoder network by trying out different combinations of these inputs as an ablation study. We investigate the impact of the choice of the inputs on the segmentation performance in previously unseen videos with regards to the recall, precision, and F1

Table 2. Comparison of method in terms of F-Measure on Change Detection 2014 Dataset

Method	Baseline	Bad weather	Camera Jitter	Dynamic Background	Int.Obj. motion	Low Framerate	Night Shadow	Thermal Turbulence	Overall		
ViBe [5]	0.90	0.53	0.66	0.22	0.26	0.60	0.62	0.67	0.58	0.58	
SubSense [10]	0.92	0.81	0.80	0.69	0.48	0.81	0.76	0.82	0.86	0.79	0.77
PAWCS [20]	0.93	0.66	0.83	0.88	0.21	0.91	0.63	0.86	0.65	0.68	0.73
IUTIS-5 [21]	0.96	0.80	0.83	0.75	0.65	0.85	0.75	0.82	0.88	0.63	0.79
DeepBS [9]	0.95	0.61	0.88	0.81	0.60	0.49	0.16	0.94	0.89	0.77	0.71
FgSegNet [12]	0.06	0.12	0.36	0.12	0.46	0.60	0.27	0.27	0.18	0.02	0.22
BSUV-Net [13]	0.97	0.88	0.76	0.77	0.59	0.96	0.82	0.92	0.87	0.41	0.80
MOS-Net(ours)	0.97	0.85	0.73	0.57	0.76	0.91	0.81	0.81	0.92	0.92	0.83
MOS-Net 2.0(ours)	0.95	0.89	0.76	0.78	0.75	0.89	0.80	0.85	0.90	0.94	0.85

score. The results of this ablation study are shown in Table 5, where the training parameters are the same for all combinations. Current Fr. is the current frame of the video, and Flux Tensor refers to the traditional solution, the flux tensor algorithm. 3D CNN refers to the 3D CNN network used in the MOS-Net architecture. On the other hand, 3D CNN + ConvLSTM indicates the spatiotemporal feature extractor in the MOS-Net 2.0 variant, which utilizes a combination of 3D CNNs and ConvLSTM.

As seen in Table 5, the current frame gives poor results when it is used alone at the input of the encoder-decoder network. Because spatial features are insufficient for moving object segmentation, temporal information must also be given to the classifier. When the current frame and flux tensor algorithm are used together, it is seen that there is a severe increase in the F1 score. The motion information obtained from the flux tensor algorithm significantly increased the performance of the segmentation network. When the 3D CNNs network output is added as an additional input to this network, an average F1 score of 0.83 is obtained. With this result, the impact of 3D CNN features on segmentation performance has been validated versus the combination of current frame and flux tensor algorithm. After 3D CNN features are added to the input of the segmentation network in addition to current frame and flux tensor, there is a massive increase of 0.29 in the F1 score.

MOS-Net 2.0 is an improved version of MOS-Net. ConvLSTM is added to the spatiotemporal feature extractor network consisting of 3D CNNs for extracting long-term spatiotemporal features. As can be seen in Table 5, the ConvLstm model brings a slight performance increase over MOS-Net. In order to show the effect of motion information formed as a result of the flux tensor algorithm, the flux tensor algorithm was removed from the inputs of the MOS-Net and MOS-Net 2.0 studies, and training was conducted. It has been observed that there is a decrease in the F1 score obtained on the test videos when the flux tensor algorithm is not present. It has been demonstrated that extracting the motion information with the flux tensor algorithm is an important input for the deep network's segmentation success on unseen videos.

6. Conclusions

In recent years, there has been a growing interest in developing methods for motion analysis and object segmentation in video sequences. These tasks are crucial for a wide range of applications, including video surveillance, traffic monitoring, and autonomous vehicle

Table 3. The qualitative results of the proposed method and comparison with the other methods



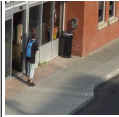



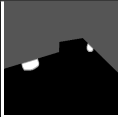
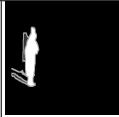
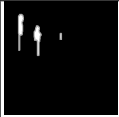

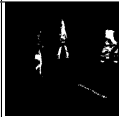


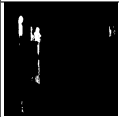



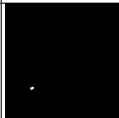







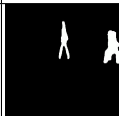

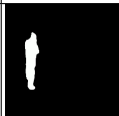




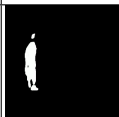





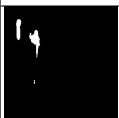

	Baseline	Night	Shadow	Thermal	Dynamic Backgr.
Current frame					
Ground Truth					
VIBE[5]					
SuBSense[10]					
PAWCS[20]					
BSUV-Net[13]					
MOS-Net(ours)					
MOS-Net 2.0 (ours)					

Table 4. The comparison of network parameter size for MOS-Net, MOS-Net 2.0, FgSegNet, and BSUV-Net

Methods	Network Size (# parameters)
FgSegNet	9,229,313
BSUV-Net	30,365,825
MOS-Net	30,382,265
MOS-Net 2.0	30,391,545

Table 5. The illustration of the impact of studies

Current Fr.	Flux Tensor	3D CNN	3D CNN+ConvLSTM	Rec	Prec	F1
✓				0.57	0.47	0.43
✓	✓			0.62	0.64	0.54
✓	✓	✓		0.82	0.85	0.83
✓	✓		✓	0.82	0.88	0.85
✓		✓		0.75	0.75	0.75
✓			✓	0.74	0.79	0.76

navigation. Traditional approaches to motion analysis and object segmentation often rely on hand-crafted features and shallow models, which can be sensitive to noise and variations in the data. To address these limitations, we have proposed MOS-Net and MOS-Net 2.0, which are based on fully convolutional encoder-decoder networks. These networks exploit the power of deep learning to learn rich, discriminative segmentation features from the data.

MOS-Net combines the flux tensor algorithm, which is a fast and efficient method for extracting motion information, with 3D convolutional filters to extract spatiotemporal features. The flux tensor algorithm operates on the principle of conservation of mass, and it has been shown to be effective for motion analysis in a variety of scenarios. By combining the flux tensor algorithm with 3D convolutional filters, MOS-Net is able to capture both motion and appearance information, which is essential for accurate object segmentation. MOS-Net 2.0 builds upon the basic architecture of MOS-Net by adding ConvLSTM layers, which are designed to capture long-term temporal dependencies in the data. By incorporating ConvLSTM layers, MOS-Net 2.0 is able to make more informed decisions based on the context of the past and future frames. Simulation experiments have shown that MOS-Net 2.0 outperforms MOS-Net in terms of both accuracy and speed.

Overall, the results of our experiments indicate that MOS-Net and MOS-Net 2.0 are effective methods for motion analysis and object segmentation. They outperform traditional approaches and recent methods such as FgSegnet and BSUV-Net, particularly in terms of F1 score when applied to unseen data. Our main contribution is the successful fusion of spatiotemporal features extracted by deep learning methods and the flux tensor algorithm, which leads to significant performance improvements. We believe that MOS-Net and MOS-Net 2.0 have the potential to make a significant impact on a wide range of applications. The code for the simulations of the proposed deep networks is available at <https://github.com/anilturker/MovingObjectSegmentation>.

References

1. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. CoRR abs/1505.04597 (2015), <http://arxiv.org/abs/1505.04597>
2. Bunyak, F., Palaniappan, K., Nath, S.K., Seetharaman, G.: Flux tensor constrained geodesic active contours with sensor fusion for persistent object tracking. J. Multimed. 2(4) (Aug 2007)

3. Turker, A., Eksioglu, E.M.: A fully convolutional encoder-decoder network for moving object segmentation. In: 2022 International Conference on INnovations in Intelligent SysTems and Applications (INISTA). pp. 1–6 (2022)
4. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149). vol. 2, pp. 246–252 Vol. 2 (1999)
5. Barnich, O., Van Droogenbroeck, M.: Vibe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing* 20(6), 1709–1724 (2011)
6. Oliver, N., Rosario, B., Pentland, A.: A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8), 831–843 (2000)
7. Braham, M., Piérard, S., Van Droogenbroeck, M.: Semantic background subtraction. In: 2017 IEEE International Conference on Image Processing (ICIP). pp. 4552–4556 (2017)
8. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. *Neural Computation* 1(4), 541–551 (1989)
9. Babae, M., Dinh, D.T., Rigoll, G.: A deep convolutional neural network for video sequence background subtraction. *Pattern Recognition* 76, 635–649 (2018), <https://www.sciencedirect.com/science/article/pii/S0031320317303928>
10. St-Charles, P.L., Bilodeau, G.A., Bergevin, R.: Subsense: A universal change detection method with local adaptive sensitivity. *IEEE Transactions on Image Processing* 24(1), 359–373 (2015)
11. Gao, Y., Cai, H., Zhang, X., Lan, L., Luo, Z.: Background subtraction via 3d convolutional neural networks. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 1271–1276 (2018)
12. Lim, L.A., Keles, H.Y.: Learning multi-scale features for foreground segmentation. *CoRR abs/1808.01477* (2018), <http://arxiv.org/abs/1808.01477>
13. Tezcan, M.O., Ishwar, P., Konrad, J.: Bsuv-net 2.0: Spatio-temporal data augmentations for video-agnostic supervised background subtraction. *CoRR abs/2101.09585* (2021), <https://arxiv.org/abs/2101.09585>
14. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th international joint conference on Artificial intelligence (IJCAI), pp. 674–679. Vancouver, British Columbia (1981)
15. Nath, S.K., Palaniappan, K.: Adaptive robust structure tensors for orientation estimation and image segmentation. In: Bebis, G., Boyle, R., Koracin, D., Parvin, B. (eds.) *Advances in Visual Computing*. pp. 445–453. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *CoRR abs/1409.1556* (2015)
17. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR abs/1502.03167* (2015), <http://arxiv.org/abs/1502.03167>
18. Wang, Y., Jodoin, P.M., Porikli, F., Konrad, J., Benezeth, Y., Ishwar, P.: Cdnet 2014: An expanded change detection benchmark dataset pp. 393–400 (2014)
19. Mandal, M., Dhar, V., Mishra, A., Vipparthi, S.K.: 3dfr: A swift 3d feature reductionist framework for scene independent change detection. *IEEE Signal Processing Letters* 26(12), 1882–1886 (dec 2019), <https://doi.org/10.1109%2F1sp.2019.2952253>
20. St-Charles, P.L., Bilodeau, G.A., Bergevin, R.: A self-adjusting approach to change detection based on background word consensus. In: 2015 IEEE Winter Conference on Applications of Computer Vision. pp. 990–997 (2015)
21. Bianco, S., Ciocca, G., Schettini, R.: Combination of video change detection algorithms by genetic programming. *IEEE Transactions on Evolutionary Computation* 21(6), 914–928 (2017)

Anil Turker is an accomplished engineer specializing in the field of Electronics and Communication Engineering. He was born on August 14, 1996, in Amasya, Turkey. Anil completed his B.Sc. degree in Electric and Electronic Engineering at Ankara Yıldırım Beyazıt University in 2019, where he achieved the top rank in his faculty. He further pursued his academic journey and obtained his M.Sc. degree in Electronics and Communication Engineering from Istanbul Technical University in 2022. Anil's research interests lie in the area of computer vision and deep learning, with a particular focus on moving object segmentation.

Ender Mete Eksioğlu received the B.Sc. and M.Sc. degrees in Electrical Engineering from the University of Michigan, Ann Arbor, in 1997 and 1999, respectively. He completed the Ph.D degree at the Istanbul Technical University (ITU) in 2005. He is currently a Professor at the Electronics and Communication Engineering department in ITU. His current research interests include deep learning for imaging problems, sparsity-based signal processing and their applications.

Received: January 29, 2023; Accepted: May 10, 2023.

Echo State Network for Features Extraction and Segmentation of Tomography Images^{*}

Petia Koprinkova-Hristova¹, Ivan Georgiev^{1,2}, Miryana Raykovska¹

¹ Institute of Information and Communication technologies, Bulgarian Academy of Sciences
Acad. G. Bonchev str. bl.2, Sofia 1113, Bulgaria

petia.koprinkova@iict.bas.bg
ivan.georgiev@parallel.bas.bg
miriana.raykovska@iict.bas.bg

² Institute of Mathematics and Informatics, Bulgarian Academy of Sciences
Acad. G. Bonchev str. bl.8, Sofia 1113, Bulgaria

Abstract. The paper proposes a novel approach for gray scale images segmentation. It is based on multiple features extraction from a single feature per image pixel, namely its intensity value, via a recurrent neural network from the reservoir computing family - Echo state network. The preliminary tests on the benchmark gray scale image Lena demonstrated that the newly extracted features - reservoir equilibrium states - reveal hidden image characteristics. In present work the developed approach was applied to a real life task for segmentation of a 3D tomography image of a bone whose aim was to explore the object's internal structure. The achieved results demonstrated the novel approach allows for clearer revealing the details of the bone internal structure thus supporting further tomography image analyses.

Keywords: reservoir computing, Echo state network, intrinsic plasticity, gray image, segmentation, tomography.

1. Introduction

Image segmentation aims division of a digital image into segments in order to reduce its complexity and thus to enable further processing or analysis. It has many practical applications including medical image analysis, computer vision for autonomous vehicles, face recognition and detection, video surveillance, satellite image analysis etc.

Image segmentation is done by assignment of labels to pixels in dependence on their belonging to individual objects in the image, such as people, animals, flowers, cars etc. Usually high-level features such as color or contrast are applied by the traditional approaches. However, these have to be fine tuned or manually assigned so the obtained results might not be accurate enough, especially for complex images.

Modern segmentation techniques relay on machine learning approaches, such as deep learning neural networks [29], for automated features extraction and segmentation. In case of gray images such as produced for medical diagnostics Computed Tomography (CT), Magnetic Resonance Imaging (MRI) etc., the problem for clearer separation of regions of interest is of crucial importance because they were often used for tumor detection or

^{*} This is an extended version of the conference paper "Reservoir Computing Approach for Gray Images Segmentation", presented at IEEE INISTA 2022.

diagnostic of brain deceases. A recent review in the area [26] concludes that all of the applied by far algorithms need broad improvement. Extensive research on gray and/or medical images includes numerous approaches such as heuristic optimization algorithms like bee colony [6] or Particle swarm [1], artificial neural networks [23, 22, 19] including deep ones [15, 14, 9, 30, 5, 25, 4, 2], gray value thresholding [21], Gaussian mixture models [18], Support vector machines [3], neuro-fuzzy approaches [24, 8] etc.

However, the most often applied DNNs need huge amount of training examples as well as big computational resources and time to be trained properly. In contrast to them Echo state networks (ESN) belong to a novel and rapidly developing family of reservoir computing approaches [7, 17] whose primary aim was development of fast trainable recurrent neural network architectures (RNN) for approximation of nonlinear time series. Following different view point to dynamic reservoir structure and its properties, in [10] a novel approach for features extraction from multidimensional data sets using ESN was proposed. It was successfully tested on numerous practical examples, among which clustering and segmentation of multi-spectral images [11].

The core of the original approach was to use the reservoir equilibrium states corresponding to each one of multidimensional input data. The fitting of the ESN reservoir dynamics to reflect the input data structure was achieved using an approach for tuning of ESN reservoir internal connectivity and dynamics called Intrinsic Plasticity (IP) [28, 27].

In [13] a modified version of the approach from [11] for segmentation of gray scale images was proposed. In order to enhance original gray image an IP tuned ESN reservoir was applied to extract multiple features from the intensity value of each pixel that are reservoir neurons equilibrium states corresponding to the pixels' intensities. The extracted features were further used by a segmentation routine to divide image into several segments (clusters). Comparison of segmentation results on the benchmark gray image Lena revealed that using the extracted via ESN features and kmeans clustering allowed to reveal hidden image details in comparison with clustering via the scalar original feature (image pixel intensity).

The present work continues investigation of our newly proposed approach applying it to a real life problem - 3D micro-CT tomography image purification via segmentation.

The paper is organized as follows: the next section 2 introduces ESN, IP tuning algorithm and adapted to gray scale images approach for features extraction; the following section presents obtained results on a set of tomography images collection; the paper finishes with concluding remarks and direction for future work.

2. ESN Approach for Gray Scale Images Features Extraction

2.1. ESN and IP tuning

The structure of an ESN is shown on Fig 1.

It incorporates a dynamic reservoir of neurons with a sigmoid activation function f^{res} (usually the hyperbolic tangent) and randomly generated recurrent connections W^{res} . The reservoir state for the current time instant k $r(k)$ depends both on its previous state $r(k-1)$ and on the current input $u(k)$ as follows:

$$r(k) = f^{res}(diag(a)W^{in}u(k) + W^{res}r(k-1) + diag(b)) \quad (1)$$

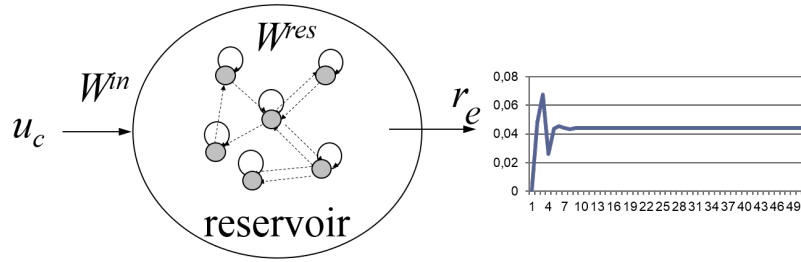


Fig. 1. ESN for features extraction

Here W^{in} and W^{res} are input and recurrent connection weight matrices that are randomly generated according to recipes given by [17]; a and b are vectors called gain and bias that are set to 1 and 0 respectively in most applications. For the aim of time series modelling the readout of reservoir is a linear combination of its current state whose parameters are the only trainable parameters of this recurrent neural network structure. However, in present investigation as in [10] we will focus only on characteristics of the reservoir equilibrium state r_e achieved after feeding of constant input u_c until the reservoir settles down (as shown on Fig. 1).

In order to adjust the reservoir to the structure of its input data, [28, 27] proposed an approach called Intrinsic Plasticity (IP) tuning that achieves desired distribution of reservoir output via changes in the gain a and bias b in equation (1). The procedure is gradient algorithm minimizing Kullback-Leibler divergence between actual and target distributions. In case of hyperbolic tangent activation function the proper target distribution is normal (Gaussian) with mean μ and variance σ :

$$p_{norm} = (1/\sigma\sqrt{2\pi})e^{-(r-\mu)^2/2\sigma^2} \tag{2}$$

Thus the training rules for IP tuning of gain and bias vectors derived in [27] are:

$$\Delta b = -\eta(-\mu/\sigma^2 + r/\sigma^2(2\sigma^2 + 1 - r^2 + \mu r)) \tag{3}$$

$$\Delta a = \eta/a + \Delta b x \tag{4}$$

Here η is learning rate and $x = W^{in}u + W^{res}r$ is the net input to the reservoir neurons. The Algorithm 1 presents the IP tuning procedure. All original data vectors (in our terminology further features) $f(i), i = 1 \div n_f$ are fed consecutively to the ESN reservoir with zero initial state $r(0) = 0$ for n_{IP} IP tuning iterations (usually 3-5 are enough as in [27]).

In [10] for the first time it was proposed to exploit the achieved by feeding of multidimensional data input equilibrium states of IP tuned ESN reservoir as new feature vector. In [11] the approach was applied to various multidimensional data sets for the aims of clustering. It was demonstrated that projections onto new (ESN equilibrium) state space could led to better data separation. The Algorithm 2 present the features extraction procedure.

Algorithm 1 IP tuning

IP tuning of ESN

2: Generate initial ESN reservoir
 $r(0) = 0$

4: $a(0) = 1$
 $b(0) = 0$

6: **for** $k = 1 \div n_{IP}$ **do**
 for $i = 1 \div n_f$ **do**

8: $x(i) = W^{in}f(i) + W^{res}r(i-1)$
 $r(i) = \tanh(a(i-1)x(i) + b(i-1))$

10: $a(i) = a(i-1) + \Delta a(i)$
 $b(i) = b(i-1) + \Delta b(i)$

12: **end for**

end for

Algorithm 2 Features extraction

Extract features from IP tuned ESN

2: **for** $i = 1 \div n_f$ **do**
 $r(0) = 0$

4: **for** $k = 1 \div n_{it}$ **do**
 $x(k) = W^{in}f(i) + W^{res}r(k-1)$
 $r(k) = \tanh(ax(k) + b)$

6: **end for**

8: $r_e^i = r(k)$

end for

All original vectors of features $f(i), i = 1 \div n_f$ are fed consecutively n_{it} times as constant input $u(k) = f(i) = const., k = 1 \div n_{it}$ to the ESN reservoir with zero initial state $r(0) = 0$ until it settles to a new equilibrium state $r_e(f(i)) = r(n_{it})$. n_{it} is number of iterations needed to achieve steady state $r_e = r(k) = r(k-1)$ of the reservoir. Achieved in this way equilibrium states of the ESN reservoir r_e^i for each original features vector $f(i)$ are considered as new features.

2.2. Approach for features extraction from gray scale images

In [10] it was investigated how the IP tuning of a randomly generated ESN reservoir led to clearer separation of the original multidimensional data after its projection to low dimensional space. The effect of IP tuning on reservoir equilibrium state was further investigated in [12] demonstrating that it increases equilibrium states memory capacity and is strongly influenced by the original data structure. Provoked by the needs of gray scale images clustering, in [13] a modification of the original approach for features extraction was proposed. In contrast to [10], from each single feature per pixel (that is its intensity value $f(i) = pi(i), i = 1 \div n_{pi}$) multiple features corresponding to ESN reservoir neurons equilibrium states r_e^i were obtained ($n_f = n_{pi}$ is the number of the gray image pixels; $pi(i)$ is i -th pixel intensity). Thus feeding of pixel by pixel intensities of an original gray image to the IP tuned ESN reservoir yields multiple filtered images as shown on Fig. 2. The obtained in this way multidimensional feature vector per pixel are subject to

clustering via kmeans using one selected feature among numerous extracted once or all of them.

3. Results and Discussion

In [13] the proposed approach was tested on the gray version of the benchmark image Lena. In present work a real life problem - 3D micro-CT tomography image purification via segmentation - was considered. Here we briefly present the results from [13] first and then continue with tomography data.

3.1. Lena Segmentation

In order to investigate effects of proposed features extraction approach a benchmark image Lena was used. The original colour image was converted to gray scale and pixels intensities p_i were scaled in range $[-1, 1]$. Next, all pixels intensities were applied to tune the gain and bias parameters of a randomly generated fully connected ESN reservoir with size $n_r = 10$ and spectral radius 0.9. The target Gaussian distribution of IP tuning was with zero mean and variance $\sigma = 0.1$ and number of IP tuning iterations was set to $n_{IP} = 5$. The number of iterations needed to achieve reservoir steady state was estimated to $n_{it} = 50$.

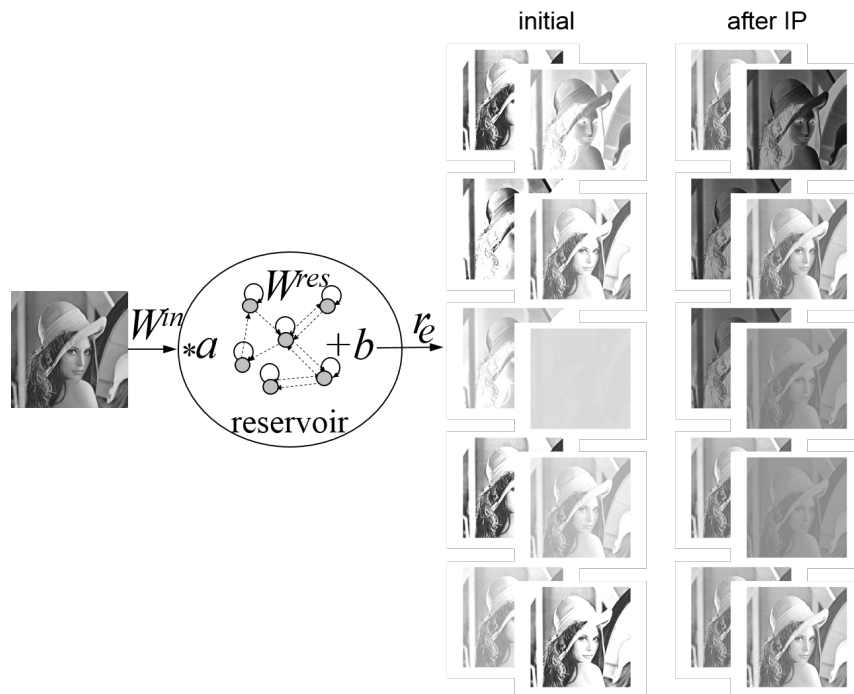


Fig. 2. Gray image features extraction

For the seek of comparison features were extracted by initial and IP tuned reservoir as shown on Fig. 2. Fig. 3 shows histograms of the original image pixels intensities vs the histograms of features extracted by initial and IP tuned ESN reservoirs.

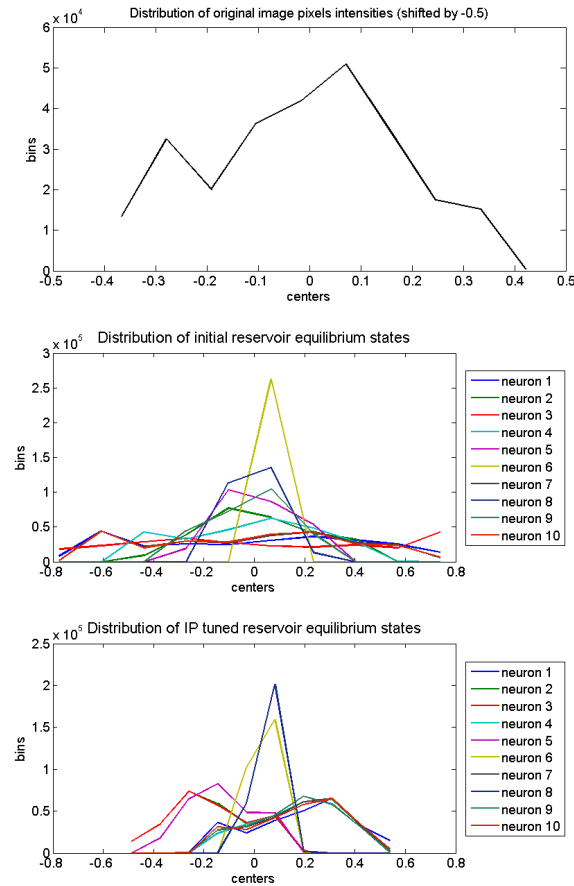


Fig. 3. Histogram of the original image pixels intensities shifted by -0.5 (top) vs the equilibrium states distributions of the initial (middle) and IP tuned reservoir (bottom)

The observed effects of IP tuning are following: the reservoir equilibrium states were squeezed in narrow interval; the distribution of new equilibrium states reflects the original data distribution.

Next kmeans clustering algorithm was applied to separate image pixels into clusters using all original and all extracted by initial and IP tuned reservoir features. From Fig. 3 is clear that features extracted from IP tuned reservoir can be visually separated into three groups - having maximum around -0.3 , 0.1 and 0.3 respectively that is in accordance other segmentation approaches tested on Lena image. So the number of clusters was set

to three here. The representative features from the three groups in Fig. 3 (neurons 1, 3 and 8) were selected for clustering too.

Comparison of the original gray image and its segmentation via kmeans clustering are shown in Fig. 4. While segmentation using features extracted by initial random ESN reservoir looks blurred, the results achieved using IP tuned reservoir reveal sharper discrimination between image regions. Segmentation via representative features (neurons 1, 3 and 8) seems quite similar to that achieved using all features extracted by IP tuned reservoir. Looking at Fig. 4, it is observed that the results achieved using features extracted by IP tuned reservoir look quite similar to those obtained by kmeans clustering of original image pixels intensities. However, IP tuned reservoir features revealed a little bit more details from the original image, e.g. some light bunches in the hair, shades on the shoulder, contours in the background stuff etc.



Fig. 4. Segmentation of the original gray image on the top into three clusters by kmeans clustering, from left to right using all features extracted by the initial reservoir, by all features extracted from the IP tuned reservoir and by the features from the representative neurons 1, 3 and 8

Comparison between results before and after IP tuning shows that fitting the reservoir to the input data allows to achieve sharper discrimination between regions with different grades of pixels intensities. Besides, in [13] it was observed that each neuron acts as a filter revealing different image characteristics.



Fig. 5. Segmentation of the original gray image into three clusters using original image pixels intensities, from left to right via hard thresholding, multi-level thresholding, fuzzy c-means, subtractive clustering and kmeans clustering

For the seek of comparison the gray image was segmented using original pixels intensities and several clustering approaches: kmeans, hard thresholding using fixed threshold levels equally distributed within range of pixels intensities, multi-level thresholding using Otsu's method [20], fuzzy c-means clustering and subtractive clustering [31]. Fig. 5 presents comparison of the original gray image and its segmentation by enumerated clustering approaches. It is obvious that kmeans segmentation outperforms all other approaches achieving sharper segmentation of the image.

3.2. Tomography Image Segmentation

The sample was scanned through a micro-CT device Nikon XT H 225. The X-ray parameters were a voltage of 100 kV and a current of 100 μ A. A Series of 2000 projections with one frame per projection, an exposure time of 500 ms, and an isotropic resolution of 97 μ m per voxel were acquired during a continuous rotation range of 360°. The images were reconstructed with CT Pro 3D (version XT 3.1.3, Nikon Metrology, Hertfordshire, UK), using a beam hardening correction and noise reduction and a median filter with a kernel size of 3X3. ROI of interest was selected and extracted as a separate volume from a human skull's central part of the sutura sagittalis.

In order to determine proper number of clusters kmeans clustering was applied to a single slice of the original tomography image. Results from segmentation into two, three and four classes are shown on Fig. 6. It is obvious that the higher the number of clusters is the more details from the gray image are discovered via segmentation.

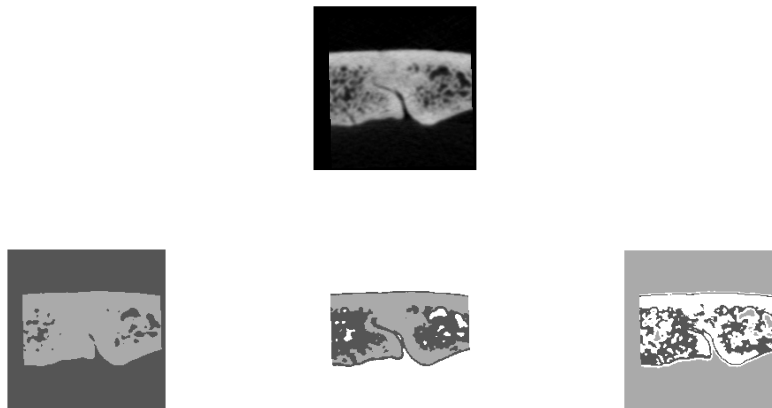


Fig. 6. Segmentation of the original image (top) into two (bottom, left), three (bottom, middle) and four (bottom, right) clusters

In order to assess the segmentation quality clusters silhouettes are compared on Fig. 7. Although the segmentation into four clusters seems more impressive on Fig. 6, its maximal silhouette value (0.9906) is slightly lower than that in the three clusters case

(0.9912). In case of two clusters the maximal silhouette value (0.9886) is the smallest one. Thus the proper number of clusters is two.

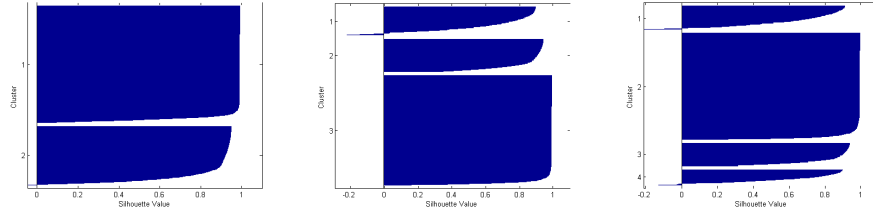


Fig. 7. Silhouettes of clusters in case of two (left), three (middle) and four (right) classes respectively

The initial random ESN reservoir with the same parameters as in previous example was IP tuned using a single slice from a different tomography collection. Next the IP tuned reservoir was applied to extract features of the investigated tomography image. Next kmeans clustering into two classes was applied using all ESN extracted multidimensional features as well as on each ESN feature separately. For the seek of comparison kmeans clustering was applied to the original image using points gray scale as a feature.

In order to asses segmentation results without ground truth information several approaches are applied [32]. Here we compare the squared error (Fig. 8) and the average squared error of the segments $F(I)$ (Fig. 9) as in [16].

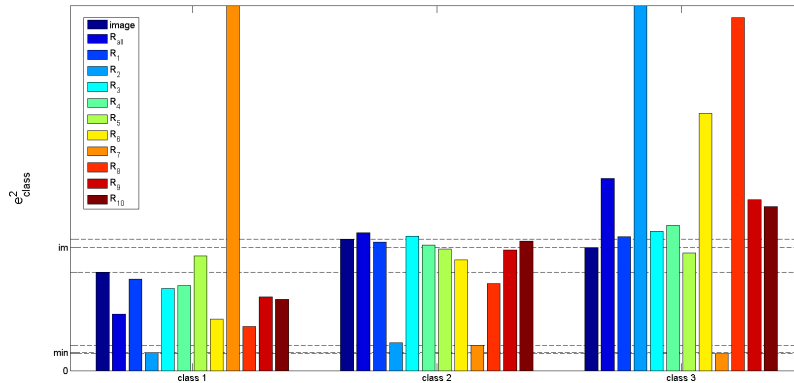


Fig. 8. Squared error per cluster

Fig. 8 reveals that using all ESN extracted features does not outperform segmentation from the original image features. However, several single ESN features give much better results for some of the segments, e.g. $R7$ that is the steady state of neuron number 7 of

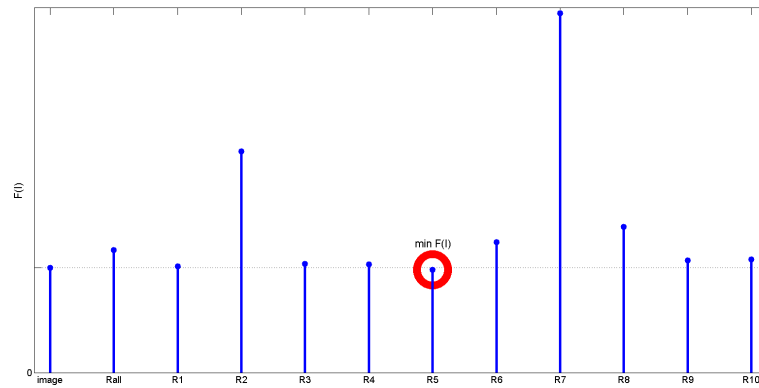


Fig. 9. Average squared error of all clusters

the ESN reservoir achieves the lowest error for classes 2 and 3 but the worst result for class 1. Fig. 9 shows that the average squared error $F(I)$ is lowest in case of feature $R5$. However, visual assessment of the segmentation results does not support this quantitative conclusion.

Fig. 10 shows entropy-based evaluation metric from [33]. It reveals that feature $R7$ is the best one based on this measure.

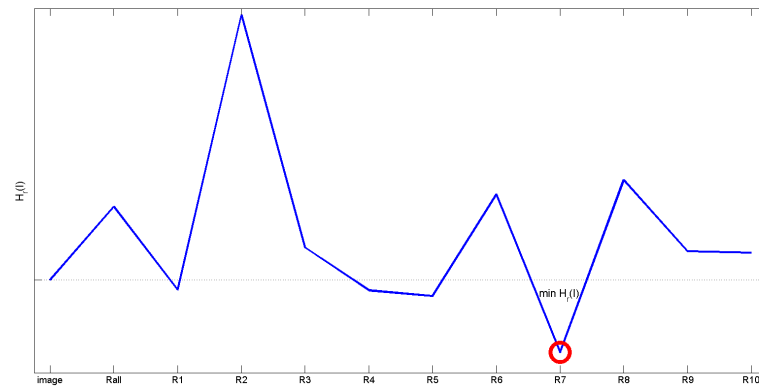


Fig. 10. Entropy evaluation metric

Fig. 11 shows the original unsegmented render (3D tomography image) and its segmentation into three classes using original features (pixels intensities) as well ESN ex-

tracted features. As expected, clustering made the image less blurred. The best segmentation was achieved by a single ESN extracted feature (from neuron 7).

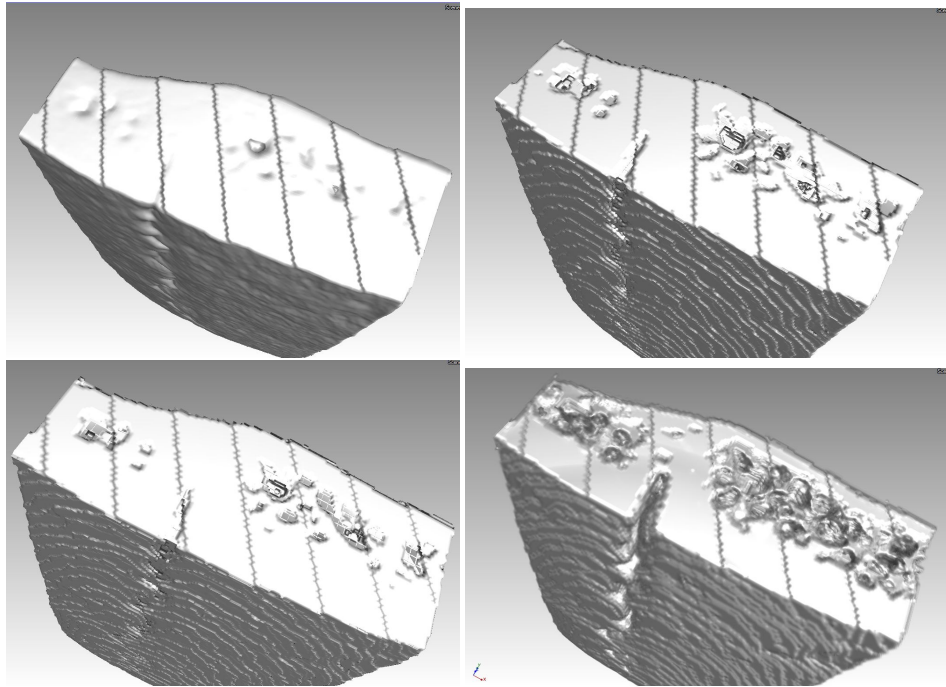


Fig. 11. Original tomography image (top, left) versus segmentation into three clusters via original image pixels intensities (top, right), all ESN extracted features (bottom, left) and best extracted feature from neuron 7 (bottom, right)

In order to explore the deeper structure of the investigated bone sample we show two representative vertical and horizontal 2D slices from the 3D image as shown on Fig. 12. Figure 13 shows the representative slices on horizontal and vertical directions from the original tomography image while at Fig. 12 while figures 14, 15 and 16 show the corresponding representative 2D slices from the segmented 3D images using original pixels intensities, all and the best ESN extracted features respectively.

While results achieved using original and all ESN extracted features look similar, the clustering via the best ESN feature reveals much more details of the tomography image. In it the compact and spongy layers of the bone are most clearly distinguished and the bone suture is fully visible in terms of placement and shape. The most impressive result is the clearest visualization of the skull seam.

4. Conclusions

Our results demonstrated that using of IP tuned ESN for features extraction from the tomography images gives numerous features among which the best ones revealed hidden

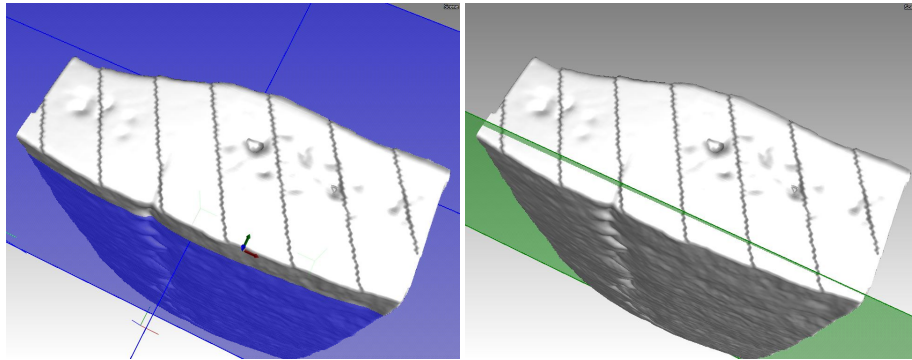


Fig. 12. Representative horizontal (blue plane) and vertical (green plane) 2D slices positions

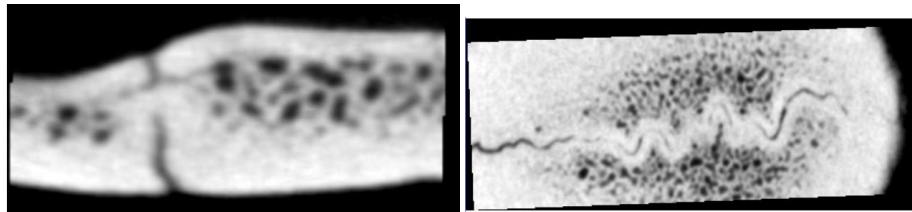


Fig. 13. 2D slices from the original tomography image

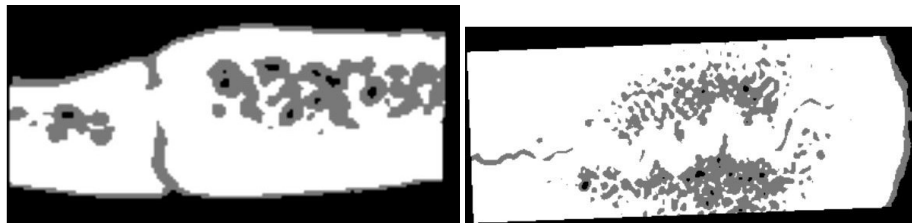


Fig. 14. 2D slices from the segmented via original tomography image pixels intensities



Fig. 15. 2D slices from the segmented tomography image via all ESN extracted features

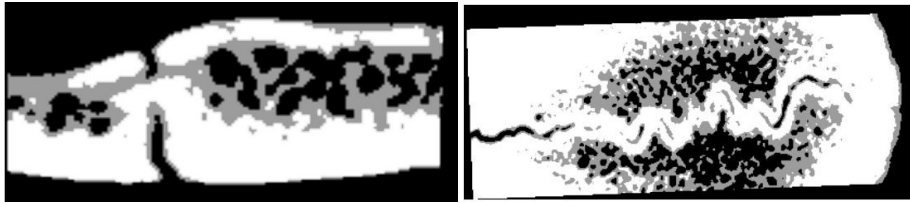


Fig. 16. 2D slices from the segmented tomography image via the best single ESN extracted feature

image details. Clustering results based on a chosen best extracted feature demonstrated impressively better segmentation of the gray image in comparison with the segmentation by its original pixels intensity.

These preliminary results are good basis for further development of hierarchical (deep) approach for gray images segmentation combining ESN and clustering approaches.

Acknowledgments. We acknowledge the provided access to the e-infrastructure of the Laboratory for 3D Digitization and Microstructure Analysis at IICT-BAS, Grant No BG05M2OP001-1.001-0003. The second author was partially supported by the Bulgarian NSF Grant KP-06-N27/6. The third author was partially supported by the Bulgarian NSF Grant KP-06-N53/7.

References

1. Alraddady, F., Zanaty, E.A., Abu bakr, A.H., Abd-Elhafiez, W.M., Fusion Strategy for Improving Medical Image Segmentation (2023) *Computers, Materials and Continua*, 74 (2), pp. 3627-3646.
2. Arutperumjothi, G., Devi, K.S., Rani, C., Srinivasan, P., Qualitative Abnormalities of Peripheral Blood Smear Images Using Deep Learning Techniques (2023) *Intelligent Automation and Soft Computing*, 35 (1), pp. 1069-1086.
3. Bao, X.-X., Zhao, C., Bao, S.-S., Rao, J.-S., Yang, Z.-Y., Li, X.-G., Recognition of necrotic regions in MRI images of chronic spinal cord injury based on superpixel (2023) *Computer Methods and Programs in Biomedicine*, 228, art. no. 107252.
4. Fang, L., Wang, X., Multi-input Unet model based on the integrated block and the aggregation connection for MRI brain tumor segmentation (2023) *Biomedical Signal Processing and Control*, 79, art. no. 104027.
5. Huang, X., Liu, Y., Li, Y., Qi, K., Gao, A., Zheng, B., Liang, D., Long, X., Deep Learning-Based Multiclass Brain Tissue Segmentation in Fetal MRIs (2023) *Sensors*, 23 (2), art. no. 655.
6. Ibrahim, S., Abu Samah, K.A.F., Hamzah, R., Ali, N.A.M., Aminuddin, R., Substantial adaptive artificial bee colony algorithm implementation for glioblastoma detection (2023) *IAES International Journal of Artificial Intelligence*, 12 (1), pp. 443-450.
7. Jaeger, H., Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach. GMD Report 159, German National Research Center for Information Technology (2002)
8. Karthikeyan, M.P., Mary Anita, E.A., IM-EDRD from Retinal Fundus Images Using Multi-Level Classification Techniques (2023) *Intelligent Automation and Soft Computing*, 35 (1), pp. 567-580.

9. Khan, R., Akbar, S., Mehmood, A., Shahid, F., Munir, K., Ilyas, N., Asif, M., Zheng, Z., A transfer learning approach for multiclass classification of Alzheimer's disease using MRI images (2023) *Frontiers in Neuroscience*, 16, art. no. 1050777.
10. Koprinkova-Hristova, P., Tontchev, N., Echo state networks for multidimensional data clustering, In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) *Int. Conf. on Artificial Neural Networks 2012*, LNCS vol. 7552, pp. 571–578. Springer, Heidelberg (2012)
11. Koprinkova-Hristova, P., Multidimensional data clustering and visualization via Echo state networks, In: Kountchev, R., Nakamatsu, K. (eds.) *New Approaches in Intelligent Image Analysis*, Intelligent Systems Reference Library vol. 108, pp. 93–122. Springer, Cham (2016)
12. Koprinkova-Hristova, P., On effects of IP improvement of ESN reservoirs for reflecting of data structure. In: *Proc. of the International Joint Conference on Neural Networks (IJCNN) 2015*, IEEE, Killarney, Ireland, DOI: 10.1109/IJCNN.2015.7280703 (2015)
13. Koprinkova-Hristova, P., Reservoir computing approach for gray images segmentation. In: *Proc. of the 2022 International Conference on INnovations in Intelligent SysTems and Applications (INISTA) 2022*, pp. 1-6, DOI: 10.1109/INISTA55318.2022.9894221 (2022)
14. Lee, J., Lee, M., Lee, J., Kim, R.E.Y., Lim, S.H., Kim, D., Fine-grained brain tissue segmentation for brain modeling of stroke patient (2023) *Computers in Biology and Medicine*, 153, art. no. 106472.
15. Lin, C.-T., Ghosh, S., Hinkley, L.B., Dale, C.L., Souza, A.C.S., Sabes, J.H., Hess, C.P., Adams, M.E., Cheung, S.W., Nagarajan, S.S., Multi-tasking deep network for tinnitus classification and severity prediction from multimodal structural MR images (2023) *Journal of Neural Engineering*, 20 (1), art. no. 016017.
16. Liu, J., Yang, Y.-H., Multi-resolution color image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16 (7), 1994, pp. 689–700.
17. Lukosevicius, M., Jaeger, H., Reservoir computing approaches to recurrent neural network training, *Computer Science Review*, vol. 3, pp. 127–149 (2009)
18. Mamalakis, M., Garg, P., Nelson, T., Lee, J., Swift, A.J., Wild, J.M., Clayton, R.H., Automatic development of 3D anatomical models of border zone and core scar regions in the left ventricle (2023) *Computerized Medical Imaging and Graphics*, 103, art. no. 102152.
19. Mustafa, S., Jaffar, A., Iqbal, M.W., Abubakar, A., Alshahrani, A.S., Alghamdi, A., Hybrid Color Texture Features Classification Through ANN for Melanoma (2023) *Intelligent Automation and Soft Computing*, 35 (2), pp. 2205-2218.
20. Otsu, N., A threshold selection method from gray-level histograms, *IEEE Trans. on SMC*, vol. 9 (1), 1979, pp.62-66.
21. Palmkron, S.B., Bergenståhl, B., Håkansson, S., Wahlgren, M., Fureby, A.M., Larsson, E., Quantification of structures in freeze-dried materials using X-ray microtomography (2023) *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, 658, art. no. 130726.
22. Pavuluri, K., Scott, J.M., Huston III, J., Ehman, R.L., Manduca, A., Jack Jr, C.R., Savica, R., Boeve, B.F., Kantarci, K., Petersen, R.C., Knopman, D.S., Murphy, M.C., Differential effect of dementia etiology on cortical stiffness as assessed by MR elastography (2023) *NeuroImage: Clinical*, 37, art. no. 103328.
23. Putri, E.R., Zarkasi, A., Prajitno, P., Soejoko, D.S., Artificial neural network for cervical abnormalities detection on computed tomography images (2023) *IAES International Journal of Artificial Intelligence*, 12 (1), pp. 171-179.
24. Rahman, J.S.U., Selvaperumal, S.K., Integrated approach of brain segmentation using neuro fuzzy k-means (2023) *Indonesian Journal of Electrical Engineering and Computer Science*, 29 (1), pp. 270-276.
25. Rajagopal, S., Thanarajan, T., Alotaibi, Y., Alghamdi, S., Brain Tumor: Hybrid Feature Extraction Based on UNet and 3DCNN (2023) *Computer Systems Science and Engineering*, 45 (2), pp. 2093-2109.

26. Ramesh, K.K.D., Kumar, G.K., Swapna, K., Datta, D., Rajest, S.S., A Review of Medical Image Segmentation Algorithms, AI Endorsed Transactions on Pervasive Health and Technology, 04 2021 - 06 2021, vol. 7, issue 27, e6
27. Schrauwen, B., Wandermann, M., Verstraeten, D., Steil, J.J., Stroobandt, D., Improving reservoirs using intrinsic plasticity, Neurocomputing, vol. 71, pp. 1159–1171 (2008)
28. Steil, J.J., Online reservoir adaptation by intrinsic plasticity for back-propagation-decoloration and echo state learning, Neural Networks, vol. 20, pp. 353–364 (2007)
29. Xu, M., Yoon, S., Fuentes, A., Park, D.S., A Comprehensive Survey of Image Augmentation Techniques for Deep Learning, Pattern Recognition, vol. 137, 2023, 109347.
30. Yadav, A.S., Kumar, S., Karetla, G.R., Cotrina-Aliaga, J.C., Arias-González, J.L., Kumar, V., Srivastava, S., Gupta, R., Ibrahim, S., Paul, R., Naik, N., Singla, B., Tatkar, N.S., A Feature Extraction Using Probabilistic Neural Network and BTFSC-Net Model with Deep Learning for Brain Tumor Classification (2023) Journal of Imaging, 9 (1), art. no. 10.
31. Yager, R., Filev, D., Generation of fuzzy rules by mountain clustering, Journal of Intelligent and Fuzzy Systems, vol. 2 (3), 1994, pp.209-219.
32. Zhang, H., Fritts, J.E., Goldman, S.A., Image segmentation evaluation: A survey of unsupervised methods, Computer Vision and Image Understanding, vol.110, 2008, pp.260–280
33. Zhang, H., Fritts, J.E., Goldman, S.A., An Entropy-based Objective Evaluation Method for Image Segmentation, Proceedings of SPIE - The International Society for Optical Engineering, January 2004, DOI: 10.1117/12.527167

Petia Koprinkova-Hristova is a Professor at the Institute of Information and Communication Technologies - Bulgarian Academy of Sciences, department of Parallel Algorithms and Machine Learning with Laboratory of Neurotechnologies. Her scientific interests are in the field of AI and particularly brain-inspired computing, neural networks and their applications in various fields such as brain functions modelling, reinforcement learning, optimization, image analyses etc.

Ivan Georgiev is an Associate Professor at the Institute of Information and Communication Technologies - Bulgarian Academy of Sciences - Head of Laboratory on 3D digitization and microstructural analysis. His scientific interests and experience are in the fields of: mathematical modeling and numerical simulations, biomedical, environmental and engineering applications. He is active also in the field of industrial computed tomography and microstructure analysis.

Miryana Raykovska is an Assistant at the Laboratory on 3D digitization and microstructural analysis, Institute of Information and Communication Technologies – Bulgarian Academy of Science. She has MSc degree in Endodontie. Her main research area is application of Micro-CT in the Medicine.

Received: January 28, 2023; Accepted: June 20, 2023.

VINIA: Voice-Enabled Intent-Based Networking for Industrial Automation*

Raul Barbosa^{1,2}, João Fonseca^{1,2}, Marco Araújo¹, and Daniel Corujo²

¹ Capgemini Engineering, Porto, Portugal

² Universidade de Aveiro and Instituto de Telecomunicações,
Aveiro, Portugal
dcorujo@av.it.pt

Abstract. Intent Based Networking (IBN) is a promising approach for automating and managing large and complex networks. Integrating Voice-enabled Virtual Assistants (VVAs) with IBN and Software Defined Networking (SDN) has improved network management efficiency and flexibility. However, there is still room for optimization improvement in installing intents in industrial scenarios. Network Orchestration Automation plays an important role within the Beyond 5G and 6G Networks, considering existing practices for orchestrating 5G Network Functions. This work presents an extended preliminary architecture for a voice-enabled IBN system called VINIA for industrial network automation. The new approach allows the configuration of more network assets (e.g., 5G networks), leveraging Network Orchestrators and Network Slice Managers, thus improving the system’s capabilities. The results provide insights into this solution’s potential benefits and limitations to enhance the automation of industrial networks’ management and orchestration procedures.

Keywords: Intent, Intent-driven Management Service, Network Slicing, Industrial Virtual Voice Assistant

1. Introduction

A novel method for managing networks called Intent Based Networking (IBN) is gaining popularity due to its ability to automate and streamline the setup and maintenance of complex networks. IBN is especially well-suited for industrial scenarios with high dependability, security, and real-time performance demands. It is a desirable solution when traditional network management techniques can be time-consuming, error-prone, and challenging to scale [1]. One of the critical components of IBN are the Network Orchestrators (NOs), software systems that automate the configuration and management of devices based on high-level business objectives and policies [2].

Integrating Voice-enabled Virtual Assistants (VVAs) provides enhanced capabilities to IBN-based systems in industrial scenarios. VVAs allow skillful network administrators

* This work is an extension of the paper presented in the INISTA 2022 conference: R. Barbosa and M. Araujo, "AI-driven Human-centric Control Interfaces for Industry 4.0 with Role-based Access", 2022 International Conference on INnovations in Intelligent SysTems and Applications (INISTA), Biarritz, France, 2022, pp. 1-6, doi: 10.1109/INISTA55318.2022.9894247

and not-so-familiar users to interact with the network using speech, providing a more intuitive and user-friendly interface for managing the network. VVA can enhance overall efficiency and reduce human error in network management. Studies show the importance of adopting VVAs and IBN for Human-AI collaboration in Information and Communication Technology (ICT) supply chains [3, 4].

The integration of VVAs and IBN with Software Defined Networking (SDN) has been the subject of recent research in industrial network automation. Our previous work [5] explored the potential of using VVAs to simplify installing intents in industrial network scenarios. However, the work only considered the applications using SDN controllers as the primary network orchestration tool. In this extension paper, we aim to broaden the scope of our previous study by exploring the NOs/Network Slice Managers (NSMs) landscape, considering the requirements for 6G networks regarding Network Management and Orchestration Automation [6–9], and the current standard development organization’s specifications and research. For that purpose, an analysis of the current NOs/NSMs is done, inferring the capability of these platforms to instantiate and control industrial networks more efficiently. As an outcome of this analysis, we present a new architecture for VINIA capable of efficiently orchestrating an industrial network through voice intents. This study is part of the work being developed in the project 6G BRAINS³: Bringing Reinforcement learning Into Radio Light Network for Massive Connections. Developing automation solutions for 6G Networks, leveraging AI/ML, and using Intent-based management by design is an important project requirement [10].

The rest of the document is structured as follows. In section 2, a study of relevant works on the topics covered in the paper is done. In section 3, we present the solution design given previously [5], extending the information about its implementation and results. Moreover, in this section, we draw some conclusions about state of the art and the drawbacks of the previous architecture to propose a more generic voice-enabled IBN system architecture that can interact with different types of SDN and Network Functions Virtualization Management and Orchestration (NFV MANO) systems. Finally, section 4 concludes and draws directions for future works.

2. Related Work

This section will review related work in speech recognition systems, IBN, and network management/orchestration to provide a foundation for our new architecture, considering current research and standardization efforts.

2.1. Speech recognition systems

Natural language access control policies can be unstructured and ambiguous; consequently, they cannot be directly implemented in an access control mechanism. In [11], a methodology is used to tackle this issue by applying linguistic analysis to parse natural language documents and annotate words to identify whether semantic arguments can be inferred from a given sentence. The authors claim that this methodology obtained results that can effectively identify access control policies with a precision of 79%.

³ 6G BRAINS is an H2020 research and innovation action supported by the European Commission Horizon 2020 Programme under grant agreement number 101017226

Due to the massive policy scale and the number of access control entities in open distributed information systems available in Industry 4.0 systems, existing management decision engines for access authorization suffer from performance bottlenecks. Several solutions have been proposed to overcome bottlenecks. In [12], a permission decision engine scheme based on a random forest algorithm to construct a vector decision classifier is presented, for which the authors claim to have achieved a permission decision accuracy of around 92.6%. In [13], a method for improving the policy decision performance by eliminating conflicts is proposed, but with poor improvement in the performance. [14] offers a k-means clustering on the access control policy set, concluding that the order of the policies in the policy set significantly impacts the permission decision efficiency. [15] proposes a permission decision optimization method based on two tree structures: match tree and combination tree. The match tree uses a binary search algorithm to search for the policy matching the access request rapidly. The combination tree evaluates the access request based on the matching procedure. In [16], the authors propose a methodology to generate sets of realistic synthetic natural language access control policies to overcome the constraints due to the lack of appropriate data.

The previous text discusses challenges and solutions related to natural language access control policies in access authorization for different information systems. It investigates methods that use linguistic analysis to understand and interpret natural language, highlighting the importance of efficient policy decision engines in Industry 4.0's open-distributed systems. Building upon this understanding, the following discussion introduces three distinct frameworks, *SUSI AI*⁴, *SEPIA Framework*⁵, and *Mycroft AI*⁶, contributing to the advancement of VVAs. These technologies offer innovative solutions for enhancing user interaction through voice and natural language while enabling the customization and control of the assistant's functionalities.

SUSI AI is an open-source VVA capable of interacting with the user through voice, using a Application Programming Interface (API). This VVA allows us to add more features to give the user greater control, allowing us to add, edit and remove skills. This type of VVA supports Linux, Android, and iOS and can be integrated with speakers and vehicles. It also allows the transformation of the user's data into JSON format and manipulation according to the user's intention.

SEPIA Framework means server-based, extensible, personal, and intelligent assistant with a Java server and a client that can run on various Android, iOS, Windows, Linux, and Mac platforms. The server is based on the REST architecture, and the clients use the HTTP protocol to communicate. Understanding natural language, dialogue management, and intention is done on the server. The client handles speech recognition and converts the voice into text, sending it to a *SEPIA* server to interpret and present the result to the user through text, which can be in JSON. Like the previously spoken technologies, *SEPIA* and the services already implemented allow us to create our commands.

The most relevant of the VVAs studied was *Mycroft AI*. This open-source VVA allows modifying, creating, and viewing code, enabling users to control the system. This type of VVA can be found on various systems, from Raspberry Pi, Windows, Android, and Mac. Besides these platforms, there are dedicated devices, Mark 1 and Mark 2. *Mycroft*

⁴ *SUSI AI*. [online] Available at: https://github.com/fossasia/susi_server

⁵ *SEPIA Framework*. [online] Available at: <https://sepia-framework.github.io/>

⁶ *Mycroft AI*. [Online]. Available: <https://mycroft.ai/>

works by intents, once awakened, the user expresses the intention to the system, and it tries to interpret the intention and find the appropriate Skill. These abilities can be installed or removed by users and can be easily updated to expand functionality. In addition to these advantages, *Mycroft* has the particularity of transforming the user's request into JSON format, which is useful for implementing it on the system.

2.2. Intent Based Networking Management

An intent is a type of policy that expresses objectives without mentioning how they are implemented and can be considered [17]: *Portable* as it can be moved between the different controller and network implementations and remain valid; *Abstract* as it must not contain any details of a specific network.

Intents are like a set of rules and services that define the criteria for access and resource use. Each rule is composed of a set of conditions and actions. The first set defines when the rule is applicable, and as soon as it is active, it generates a set of actions to be implemented. Policies Based Networking Management works as a manager that separates the rules that control the system from its functionality. As a result, this system reduces maintenance costs and improves run-time flexibility. The research currently being performed into Beyond 5G/6G Networks validates the interest in this concept [18–20, 7]. We have focused on studying the application of intents in two different technologies: SDN and NFV MANO.

Intent-Based Networking Management in SDN Research work by the author Vijay Varadharajan [21] explores an architecture based on policies in SDN context using the ONOS controller. This architecture is run on the SDN controller as an application. The policy servers connected to the respective controller have five main components: Repositories, Policy Managers, Policy Evaluation Engine, Policy Enforcer, and Handle Creator. In Machado's work [22], the author introduces a policy authorization framework for SDN using a high-level language. This study focuses on policies targeting Quality of Service (QoS). By interpreting Service Level Agreements (SLAs), the system extracts Service Level Objectives (SLOs), which are considered requirements for assessing the network's performance. Douglas Comer and Adib Rastegarnia presented a framework, Open Software Defined Framework (OSDF) [23], which offers a high-level API that allows network users to configure and monitor the network to provide QoS. In addition, OSDF has mechanisms to analyze conflicts between policies to prevent two or more policies from conflicting when applied to the same targets. This framework's architecture consists of four components: Policy Storage Module, Policy Conflict Detection Module, Policy Parser Module, and High-level network operation services. The main focus of this study was the analysis of conflicts since it is essential to maintain the excellent performance of the networks.

Intent-Based Networking Management in ETSI MANO Platforms This work focuses on transitioning from a standalone SDN solution implemented in the previous work, augmenting it with capabilities to control End-to-End (E2E) NOs like ONAP⁷ or OSM⁸.

⁷ ONAP.[online] Available at: <https://www.onap.org/>

⁸ OSM.[online] Available at:<https://osm.etsi.org/>

These orchestrators bring several benefits to organizations managing complex networks: Improved Scalability, Enhanced Automation, Centralized Management, Increased Flexibility, and Improved Security.

ONAP supports Intents and can perform operations for quickly deploying an E2E Network that entails 5G Core and RAN Network Functions. Moreover, the platform allows for automatic operations through the closed-loop automation associated with efficient monitoring of resources. However, the complexity of this orchestrator limits its use, especially if the required computing resources aren't available.

OSM, an Open Source alternative developed by the European Telecommunications Standards Institute (ETSI), isn't capable of the same features, as it is a less complex/able NSM. However, since it is also a NO compliant with ETSI MANO Architecture, its adoption in European-funded projects (mainly Horizon programmes) is relevant. Several solutions build on top of OSM to automate the Network Management and Orchestration procedures [20, 24–27] due to its simplicity and low computing resources requirement. Moreover, in the ETSI OSM-hosted Hackfests Ecosystem Days, several demos of work unpublished/unreferenced in the literature are performed⁹.

Telecom organizations can increase their networks' effectiveness and dependability while lowering operational expenses and enhancing security, augmenting existing SDN solutions with up-to-date NOs. NOs can deal with the requirements of contemporary networks, whether you are in charge of a sizable enterprise network or a network that supports vital infrastructure. The trend moves to more autonomous networks also brought new requirements to NOs, like the ability to manage E2E Network Slices. This led existent NOs to gain Network Slice Management capabilities [28, 29].

Standard Development Organizations Specifications and Research Automating procedures related to communication service configuration has been a research topic for several years. The need for specialized people to configure the network stacks has become a point of failure for both Communication Service Providers (CSPs) and Network Operators (NOPs) [30–32]. This has led the Telecommunication Industry to devise solutions to this problem. The 3rd Generation Partnership Project (3GPP) has been developing some research on Intent driven Management Service (Intent driven MnS) for mobile networks in the scope of the Technical Report (TR) 28.812 [33]. This effort, part of 3GPP release 17, offers an overview of the business requirements for adopting intents at different roles of the Telecom Vertical Industry. The document aligns with the recent works developed in the area of 5G, which capitalize on the concept of Network Slicing, a characteristic and a requirement of the architecture of 5G networks. The concept of management of Network Slices is one of the key points that mostly influences the TR. Specifically, 3GPP establishes that one or more Network Slices can deliver each Communication Service (CS). Starting from this concept, the definition for an Intent driven MnS specifies that it is a Service whose capabilities can be defined and managed via an Intent. 3GPP specified the consumers of the Intent as the CSP or Communication Service Consumer (CSC) building on top of the Network Slicing as a Service (NSaaS) concept defined in [34], allowing the CSP to provide management services to the CSC. This new paradigm part defined in 3GPP release 15 allows managing the resources associated with that slice. In the case of

⁹ As an example, in the OSM-MR10, the author of [24] presented a Demo called Vertical's intent evolution at service runtime driving vCDN automated scaling

a Intent driven MnS, the CSC needs to know a service’s characteristics without knowing the Network Slice’s components. The 3GPP also specifies how CSP and NPs can also leverage the use of Intents for the management of the Network Slices/Infrastructure in their domains. 3GPP’s TR 28.812 also presents how closed-loop mechanisms can be used to automate tasks leveraging the Intent for effortless control of the CS to be provided to the CSC. It’s important to reiterate that an intent specifies the CSC’s objective, detailing the expected characteristics.

While the 3GPP has focused mainly on research that builds on the most recent advancements/needs of mobile networks, other Standard Development Organizations (SDOs) proposed using Intent for other use cases. TeleManagement Forum (TM Forum) introduced Intents in the scope of the Autonomous Networks project, which has created documents that detail the technical and business characteristics of an Autonomous Network [35–38]. The Introductory Guides of this organization explain the importance of this technology to different Vertical Industries [35, 39]. These documents facilitate understanding of the concept by people unfamiliar with network management and orchestration.

Other relevant research related to IBN can be visualized in Table 1.

Table 1. Description of IBN architecture scope and related works

Platforms	Functions / Features	References
SDN	Service model and orchestration	[40],[41],[42]
	Monitoring and resource exposure	[43], [44]
	Intent deployment and configuration	[45]
	Network orchestration	[46]
NFV	Service model and orchestration	[47]
	Intent deployment and configuration	[45]
	Network orchestration	[48]

2.3. Summary

While existing speech recognition systems and intent-based networking management approaches have made significant strides, notable gaps and limitations still require attention.

One major limitation in speech recognition systems is handling highly unstructured and ambiguous natural language access control policies. While methods like linguistic analysis have been proposed to parse and annotate words in natural language documents, achieving high precision in identifying access control policies remains challenging. The existing decision engines for access authorization may suffer from performance bottlenecks due to the massive policy scale and the number of access control entities in open distributed information systems. There’s room for innovation focused on the bottlenecks, improving decision accuracy, and performance enhancement.

Regarding IBN management, existing implementations in SDN and Network Functions Virtualization (NFV) platforms have shown promise but face limitations. In SDN, while architectures based on policies have been explored, analyzing conflicts between policies remains a critical challenge to maintaining network performance. Furthermore,

SDN solutions implemented with specific controllers may lack the scalability and flexibility needed to handle complex networks effectively. However, NFV platforms and NSMs have different capabilities and complexities, leading to trade-offs in their adoption. ONAP supports Intents and can automate E2E network deployment, but its complexity may hinder its use, particularly in resource-constrained environments. OSM is more suitable for specific scenarios due to its simplicity and lower computing resource requirements, making it the NSM of choice in EU-funded research projects. However, its capabilities may not match those of ONAP.

The heterogeneity of solutions and the significant gaps and limitations analyzed led us to our architecture proposal. Using a similar approach to the one proposed by SlimANO [20], where it interfaces with NFV MANO platforms like ONAP and OSM to enhance scalability, automation, and centralized management while enabling Intent-Driven Mobile Network Slices. Moreover, running on top of both orchestrators creates a more efficient, flexible, and effective network management system that can handle the requirements of contemporary networks, including those of 6G Networks. VINIA architecture's focus on Intent-Driven Mobile Network Slices aligns with the ongoing research efforts in 3GPP, emphasizing the importance of managing Network Slices in meeting specific Communication Service requirements. Additionally, the architecture aims to overcome the trade-offs between different approaches by integrating diverse capabilities from both ONAP and OSM, offering telecom organizations more options for effectively managing their networks while considering resource constraints and complexity. Combining VVAs with Intent-Based Networking in the proposed architecture also simplifies and facilitates intent installation, further enhancing network management.

3. System Design

This chapter will present the VINIA system, detailing the characteristics and respective descriptions, system workflow, and architectures. The system is divided into two components; the first is responsible for identifying the user, and the second implements the user's intention after the permissions validation.

3.1. Speech recognition system

This section outlines the design and characteristics of the system for successful development. The system employs pre-trained voice recognition models to identify users. User recognition is divided into training and test phases, differing mainly in the data processing step. The training phase involves data sent to modeling algorithms for training, while the test phase employs an already trained model for data classification. To provide an in-depth understanding of both phases and highlight their shared components, Figure 1 illustrates a detailed workflow.

1. **Data collection**-This is the initial process of collecting user information, namely the number of speeches they have and a list corresponding to the speeches' directories.
2. **Processing phase**-After loading the speech file, four approaches are applied: a) Trimming; b) Optionally, and if necessary, Data Augmentation; c) Feature Extraction; d)

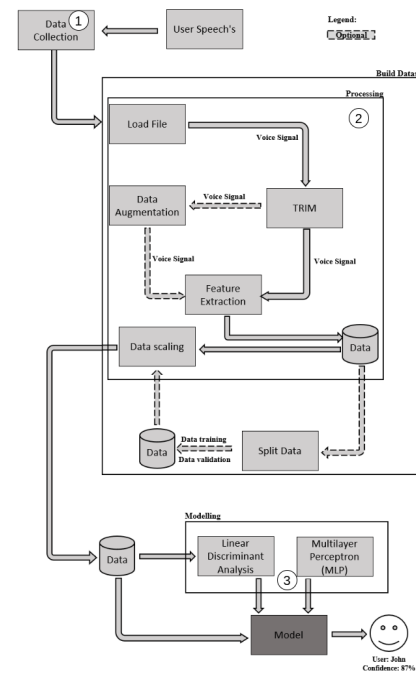


Fig. 1. VINIA's Speaker Recognition System: Workflow

Depending on the used algorithm, Multilayer Perceptron (MLP) or Linear Discriminant Analysis (LDA), optionally, we can split the Dataset in two: Data Training and Data Validation

- Modelling or testing**-We can follow two approaches, one for the modeling where the model is trained for future identification of users or for the pre-trained model where the data will be classified, that is, training and testing phases, respectively.

Regarding data construction (processing), we apply Trim, Data Augmentation, Feature Extraction, Data Scaling, and Split Data. We use Neural Network- MLP and LDA for modeling. Our choice of employing the MLP and LDA as our primary modeling techniques is based on their ample amount of work employing such techniques [49–53]. Our selection of MLP for modeling plays a pivotal role in our approach for several compelling reasons:

- It's a universal approximator, meaning it can effectively capture complex relationships within our data. This is of paramount importance when dealing with diverse and intricate audio data;
- In audio processing, extracting salient features may not be obvious to the human ear but can be efficiently identified by the model. The MLP empowers our system to uncover and use these data patterns;
- Complex tasks such as audio-based user identification using deep neural networks, including MLPs with multiple layers, require this method;

While the MLP constitutes the primary workhorse of our modeling approach, we have also complemented it with LDA for the following reasons:

- Discriminative Analysis: LDA is a proven technique for enhancing the separability of classes in data. This is particularly useful in user identification, where distinguishing between different users is a core objective;
- Dimensionality Reduction: LDA provides an avenue for dimensionality reduction, crucial in simplifying complex data while retaining essential discriminative information;

In conclusion, combining the MLP and LDA offers a balanced and comprehensive modeling approach. Together, these models qualify our system to effectively construct and process data for robust user identification, ensuring the accurate recognition of users even from complex audio data.

In the context of speech signal processing, it is crucial to discuss several key aspects involved in data construction, as these components lay the foundation for successful user identification models. These aspects include data trimming, data augmentation, feature extraction, and data scaling. Each plays a unique role in preparing the data for model training, ensuring that the system can effectively handle a wide range of scenarios. The data trimming process is designed to remove silent portions of an audio signal, particularly segments without speech. By eliminating these silent intervals, the overall duration of the speech signal is impacted. The method involves setting a threshold to identify silence zones, therefore segmenting the audio where the sound falls below 20 decibels. These identified areas are then labeled as silent zones, indicating the absence of speech. In machine learning, insufficient data for training models is a prevalent issue, especially when dealing with small datasets. To address this challenge, various data augmentation techniques can be employed, tailored to the specific nature of the data at hand. Data augmentation involves generating additional data artificially from the existing dataset. It is an optional process that is selectively employed based on the specific data requirements. When working with voice signals, which is our focus in this study, data augmentation techniques aim to create additional voice samples that expand the dataset. Common approaches for voice data augmentation include injecting noise and modifying pitch and speed. However, due to the context of voice recognition and the need to preserve the essential voice characteristics, the primary technique used here is noise injection. In this process, the original voice signal is analyzed to calculate its speech duration. Subsequently, artificial noise is generated and integrated into the original signal, effectively duplicating the data. This augmentation technique enhances the robustness and generalization of machine learning models trained on limited voice data. Feature extraction plays a pivotal role in voice signal processing, enabling the extraction of essential characteristics from the voice data. Features, in the context of voice signals, represent distinctive attributes that are crucial for voice analysis. These features can be broadly categorized into two types: temporal (time-domain features) and spectral (frequency-based features). Temporal features, which fall under the time-domain category, are relatively straightforward to extract and have direct physical interpretations. They include signal energy, zero-crossing rate, maximum amplitude, and minimum energy. These features provide insights into the temporal aspects of the voice signal and are important for tasks such as pitch and rhythm analysis. On the other hand, spectral features involve transforming the time-based voice

signal into the frequency domain using mathematical techniques like the Fourier Transform. Spectral features encompass a range of attributes, including fundamental frequency, frequency components, spectral centroid, spectral flux, spectral density, and spectral roll-off. These features are invaluable for analyzing the frequency content of voice signals, contributing to tasks related to note recognition, pitch analysis, rhythm, and melody extraction. In our feature extraction process, we employ a comprehensive set of methods that cover both temporal and spectral domains. This diverse set of techniques allows us to capture various voice signal characteristics. Our selection includes Root-Mean-Square and Zero Crossing Rate for temporal analysis, which provides insights into the signal's energy and temporal dynamics. On the spectral side, we leverage various techniques such as Chroma features, Mel-Frequency Cepstral Coefficients (MFCC), Spectral Centroid, Spectral Contrast, and Spectral Rolloff. These spectral features enable us to delve into the frequency domain aspects of the voice signals, uncovering vital information about the voice, including pitch, tone, and spectral content. In the final step of our data processing pipeline, data scaling is applied to prepare the extracted features for model training. Data scaling aims to ensure that the feature values are transformed and exhibit consistent statistical properties. We employ the Standard Scaling approach, standardizing the data's distribution to have a mean value of zero and a standard deviation of one. This step aids in optimizing the performance and convergence of machine learning models, ensuring that features with different scales do not unduly influence the learning process. All these steps were validated in the paper that this journal extends [5]. All this voice processing and model creation is integrated into an API Representational State Transfer (REST) server that stores the machine learning model to identify users. So, until now, all the techniques for building the model and working on the voice have been presented. However, it is necessary to capture the user's voice; for that, the open-source VVA *Mycroft* is used, as shown in Figure 2.

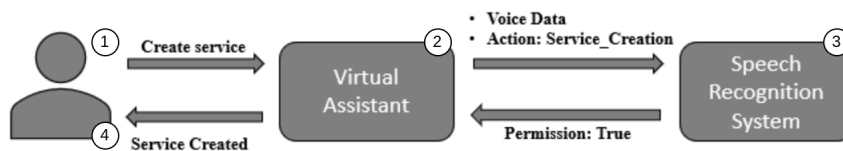


Fig. 2. Integration with Virtual Assistant

Here, the user (1) requests the creation of a service to *Mycroft* (2), which interprets the type of action required and sends it to the recognition server (3). The server recognizes the user by sending the signal captured to the machine learning model. After recognition, a query is made to the database to understand the user's roles and respective permissions. If the user has permission, the service is implemented (4). Otherwise, the user is informed

that he has no permission. The system sequence diagram that illustrates the recognition and authorization process is shown in Figure 3.

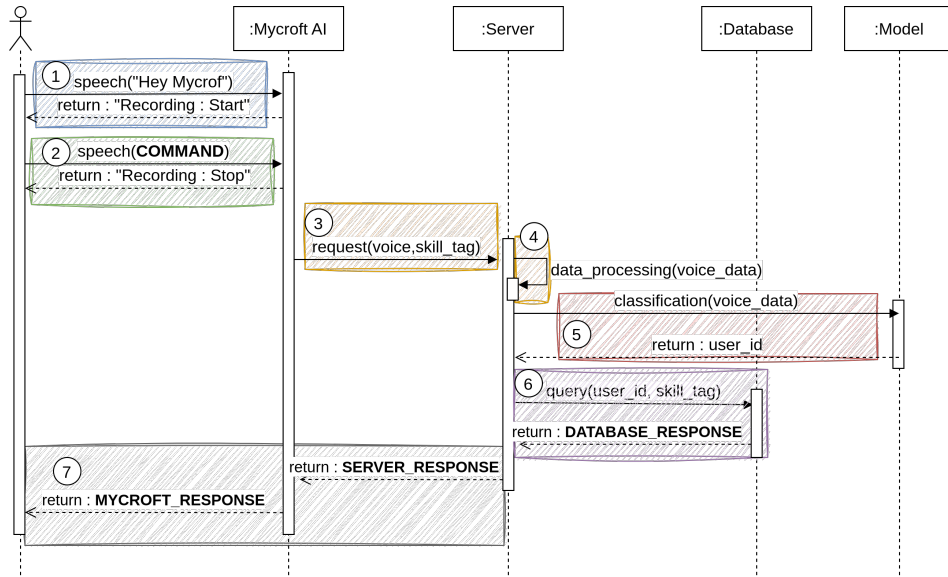


Fig. 3. Sequence System Diagram: Recognition and Authorization process

Detailing the sequence diagram, in (1), the user activates the VVAs using their default wake sentence *Hey Mycroft*. After being alerted, the recording of his command starts through a sound; the user communicates the desired command (2). The order is received on *Mycroft* and sent to the server via HTTP (3). The server receives the voice data and respective tag identifying which skill was triggered (user’s action) (3). Next, voice data is processed (trim, feature extraction, and scaling) (4) and sent to the pre-trained model to identify (5). After that, a database query is performed to fetch information about the user to verify if the respective user has permission for the action required (6). Finally, based on previous decisions, the VVA *Mycroft* responds to the user about their permission (7).

3.2. Intent Based Network

Once the user is identified and allowed to implement the request, it goes to the IBN system, where a layer of intelligence is applied to the network, replacing manual network configuration processes and reducing the complexity of creating, managing, and enforcing network policies. To understand how this part of the system will work, the VINIA’s IBN system architecture is shown in Figure 4.

The system architecture is designed to operate in a business context, where the user interacts with the VVA using their equipment (e.g., a computer). The VVA plays a pivotal role in maintaining communication between the system and the user to capture the user’s commands. After being activated, the VVA saves the user’s intention and forwards it to the

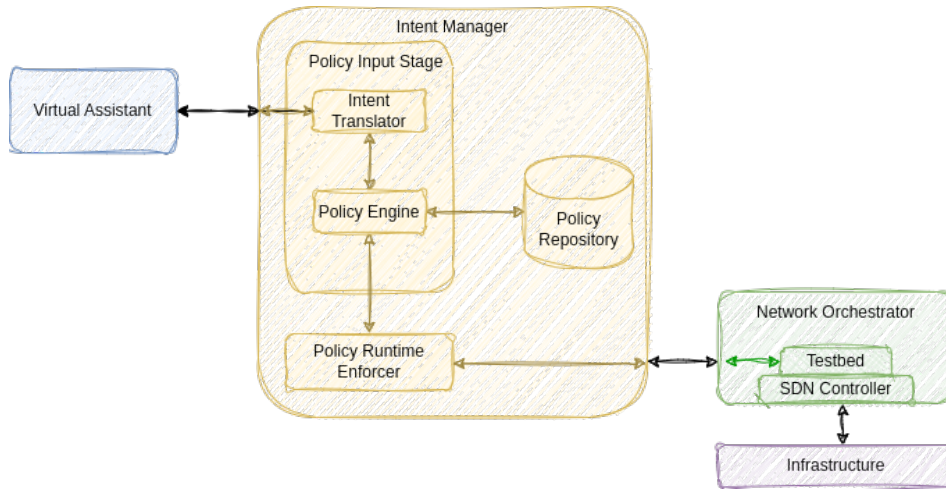


Fig. 4. VINIA's IBN System Architecture

Intent Manager to validate and implement it. The assistant is responsible for associating the policy with the user's intention and converting the high-level language into a machine language, sending it to the component where it will be analyzed.

The Intent Manager is a core system component, encompassing the Policy Input Stage, Policy Repository, and Policy Runtime Enforcer. The Intent Translator and Policy Engine components reside within the Policy Input Stage. The Intent Translator translates user intents from the VVA or other sources into a machine-readable format. The Policy Engine takes these translated intents and implements the logic and decision-making based on the policies. It ensures smooth communication between the VVA and the Policy Repository while also being accountable for analyzing policy conflicts to prevent clashes with existing network rules. In case of successful policy validation, this component forwards the rules to be implemented in the network infrastructure via Policy Runtime Enforcer. However, it initiates negotiation with the user in a conflict to resolve the issue.

The Policy Runtime Enforcer is a vigilant network monitor, continuously collecting and analyzing network information. It is critical in planning and executing policy implementations, considering operations and resources. The Policy Runtime Enforcer utilizes the data analysis results from the Policy Input Stage thresholds to take appropriate action if a policy fails.

The Testbed component is a REST server facilitating communication between the system and the SDN controller. It receives installation requests from the Policy Runtime Enforcer and attempts to install them in the SDN controller. Additionally, Testbed handles data collection requests that aid in applying monitoring algorithms and calculating variables for further study.

The SDN Controller obtains data from the network and implements any necessary rules. It communicates with the infrastructure, receiving essential information required for

the system to make the desired changes. In this architecture, the OpenDayLight (ODL)¹⁰ controller is utilized as the SDN controller.

The structure of intents represents all the information from the user structured so the system can interpret it. The system may receive several user requests, and creating a design that covers as much varied information as possible is crucial. The Intent structure template of our approach is shown in Listing 1.1.

Listing 1.1. Intent Structure

```
{
- IntentType:
- Intent Target:
- Intent State:
  Conditions:[
    Policy Type:
    Constraints:[
      Domains:[
        - Traffic Type:
        - Period:
        - Name:
        - Bool:
        - Access:
        - Performance:
      ]
    ]
  ]
}
```

Each request is assigned a type (for example: indicates whether the request is to set priorities or create services), a target (for example: what type of network), and a state (based on the state machine). Besides these three points, policies are defined according to the user’s wishes. For each user-defined policy, we can have restrictions that indicate performance values, type of traffic, and who accesses the network, among others.

VINIA’s Intent State Machine was drawn, Figure 5, considering the architecture and intent structure. It allows us to understand the life cycle of intent since it was captured by the VVA until its installation, not compromising the network operation. The intent state machine is divided into six sections. The validation phase and conflict analysis are part of the Policy input stage. The compilation and monitoring phases are processed in the Policy Runtime Enforcer module, and finally, the installation phase is the responsibility of the REST server *Testbed*.

In the initial phase, user requests are validated to ensure they include all required information for the desired action. If essential details are missing or inaccurate, an ”invalid” status is assigned, necessitating user interaction. Conversely, a ”valid” status is assigned if all necessary information is provided, allowing progression to the next phase.

The conflict phase is the second checkpoint in the state machine, comparing pre-validated attempts with stored requests in the database. Conflicts arise if requests are redundant or contradictory to existing information. When conflicts occur, a conflict state is assigned for user resolution. If there are no conflicts, the process advances to the compilation section.

During the compilation phase, the system translates user-requested policies from a high-level language into a format the controller interprets for installation. If the controller cannot support the desired information or operations, a compilation error occurs, which

¹⁰ OpenDaylight Project — An Open Platform for Network Programmability. [online] Available at: <https://www.opendaylight.org/>

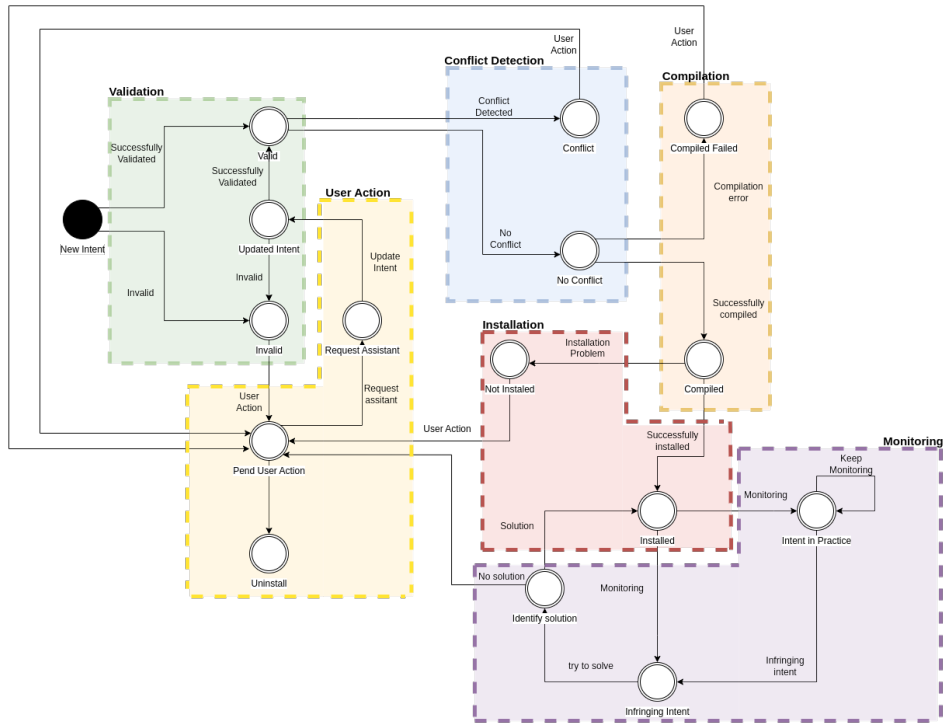


Fig. 5. VINIA's Intent State Machine

can be resolved with user assistance. Successful compilation leads to the subsequent installation phase.

Once the rules are obtained, the next step is to install them on the controller. If the controller supports the objectives and is installed, we move on to the monitoring phase. If the goals are not achievable because they may be offline or non-existent, the user is alerted that the application was not installed.

In the final phase, monitoring consists of a cycle that constantly checks if the existing requests in the database are being fulfilled or if any violation has occurred. If non-compliance occurs, the system automatically tries to identify how to solve the problem without human intervention. The user is alerted if the system does not specify a solution that corrects the problem. After the user is recognized and has permission, if the intent passes all these stages, the application is installed on the network and starts to be monitored.

3.3. Experimental Results

The results will be presented in two parts, the first referring to the speech recognition system results and the second to the intents system results.

Speech Recognition System To obtain suitable data for our study, it was necessary to collect datasets containing user speech. We utilized four distinct datasets, including NOIZEUS [54], TIMIT [55], LibrisSpeech ASR [56], and a custom dataset named SAFC.

In Table 2, these datasets are characterized by key attributes: dataset size, the number of speeches made by each user, and the average speech time.

Table 2. Dataset Properties

Name	Size (users)	Number of Speeches	Average Speech Time
NOIZEUS	6	5	2.6680 seconds
TIMIT	462	8	2.6726 seconds
LibrisSpeech	156	77 (average)	6.8771 seconds
SAFC	25	5	3.1416 seconds

As evident in Table 2, these datasets exhibit diverse characteristics, setting the stage for studying the scalability performance of the two selected algorithms concerning the growth in the number of users. This diversity also allows us to evaluate the performance of the selected approaches with respect to the amount of data per user.

In the context of industrial applications, where numerous users may be involved, it is imperative to ensure that a system can effectively handle a substantial user base. Scalability, measured as the performance of the different selected algorithms concerning the growth in the number of users, is of paramount importance. To meet this need, we have chosen "accuracy" as the primary performance metric for the study. Accuracy, in this context, serves as a critical metric due to its real-world relevance. In industrial scenarios, a high degree of user identification accuracy is vital to ensure the system functions correctly and securely. The primary objective of this study is to assess how well our selected algorithms perform in identifying users accurately, especially as the number of users grows. Accuracy reflects the system's ability to correctly classify and identify users in a multi-user environment, a crucial requirement in many practical applications. Hence, the choice of accuracy as a key metric is aligned with the practical context and the core objective of our study, which is to develop a robust and scalable user identification system suitable for real-world industrial applications. The primary study measures the performance of the different selected algorithms as a function of the growth in the number of users. Its performance is also analyzed with and without the use of data augmentation. The results are presented in Table 3, regarding the percentage of success rates.

We can conclude that both approaches perform well, and data augmentation is unnecessary in a system with minimal users and speeches per user. Regarding the experiments performed with the TIMIT dataset, we can conclude that both algorithms scale well with the increase of users, and data augmentation positively influences the performance of neural networks. Finally, to analyze the relation of the number of speeches per user, a final test of the LibrisSpeech dataset is carried out. This has 156 users; however, each has an average of 77 speeches. It is also necessary to be aware that each speech, on average, has approximately twice the work of the speeches of the other datasets. With these results, we observed that data augmentation has a negative performance on the LDA and a positive one on the neural network. This indicates that increased data per user influences the LDA performance negatively and the neural network positively.

Table 3. Dataset results

Dataset	Models	with Data Augmentation	without Data Augmentation
NOIZEUS	MLP	100%	100%
	LDA	100%	100%
TIMIT	MLP	100%	87,88%
	LDA	92,21%	95,67%
LibrisSpeech	MLP	100%	98,63%
	LDA	93,15%	93,84%
SAFC	MLP	100%	100%
	LDA	100%	100%

Intent System An illustrative scenario centered around service creation was formulated to showcase the outcomes related to the intention system. This scenario involved configuring a network topology comprising three switches and seven hosts, as depicted in Figure 6. Once created, there are no active services, as can be seen in Figure 7.

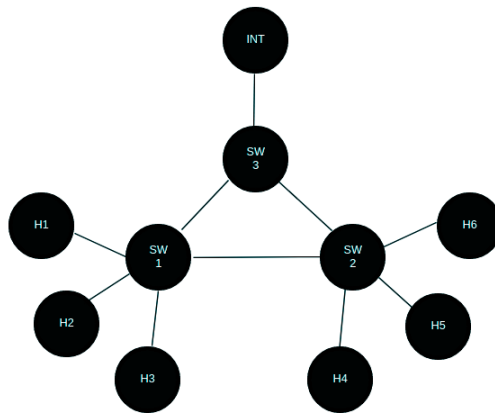


Fig. 6. Testing Environment

```

sudo ip netns exec b2_NS6 iperf3 -c 192.168.2.1 -p 4001 --cport 4006
iperf3: error - unable to connect to server: No route to host
  
```

Fig. 7. Connection test between Host 1 and 6

The creation of a service depends on four parameters: the name of the service, who accesses the service, if the service needs an Internet connection, and finally, if there is a need to limit the bandwidth of the service. To understand how the user communicates these actions to the VVA, Mycroft, the subsequent dialogue is presented:

- User: "Hey Mycroft, create a service, please."
- Mycroft: "Sure, what is the name of the service?"
- User: "Call it s1."
- Mycroft: "Ok, please name the machines that will have access to the service."
- User: "Provides access to users 1 and 6."
- Mycroft: "Regarding internet access, does the service need internet access?"
- User: "Yes, please."
- Mycroft: "One last question, do you want to define the performance of the service?"
- User: "Yes, 4000 Mbits/sec."
- Mycroft: "Thanks for the information, wait a bit while I implement the service."

The user provides information to the VVA, which then initiates a state machine presented to validate, analyze conflicts, compile, and install the intent within the network upon obtaining permission. In the provided instance, the validation phase assesses whether complete data for service creation exists, verifying valid information in all mandatory fields required. Next, during the conflict analysis phase, the system checks if any information on this new service conflicts with any already implemented. It checks if the service name exists and if the hosts involved are active in other services. If any of these cases occur, the user is alerted. Once validated and without conflicts, the request goes to the build phase. In this phase, several requests are made to both the Testbed Server and the database to convert into rules, the policies from the user. If any requests fail, the compilation phase fails, and the user needs to be alerted. We move on to the installation phase when we get the rules compiled. In this phase, the rules are sent to the Testbed Server, which communicates with ODL to install them. After they are installed, they are stored in the database, and the user is notified that the application has been installed.

To test whether the service has been created between users 1 and 6, with Internet access and bandwidth limit set, we again use the `Iperf` tool in the same context mentioned above. The connection test can be visualized in Figure 8.

```

Connecting to host 192.168.2.1, port 4001
[ 5] local 192.168.2.6 port 4006 connected to 192.168.2.1 port 4001
[ ID] Interval      Transfer      Bitrate      Retr  Cwnd
[ 5] 0.00-1.00    sec  1.62 MBytes  13.6 Mbits/sec  115  5.66 KBytes
[ 5] 1.00-2.00    sec  573 KBytes  4.69 Mbits/sec   77  4.24 KBytes
[ 5] 2.00-3.00    sec  445 KBytes  3.65 Mbits/sec   63  7.07 KBytes
[ 5] 3.00-4.00    sec  445 KBytes  3.65 Mbits/sec   74  18.4 KBytes
[ 5] 4.00-5.00    sec  509 KBytes  4.17 Mbits/sec   74  2.83 KBytes
[ 5] 5.00-6.00    sec  382 KBytes  3.13 Mbits/sec   60  4.24 KBytes
[ 5] 6.00-7.00    sec  636 KBytes  5.21 Mbits/sec   83  5.66 KBytes
[ 5] 7.00-8.00    sec  318 KBytes  2.61 Mbits/sec   69  2.83 KBytes
[ 5] 8.00-9.00    sec  573 KBytes  4.69 Mbits/sec   73  1.41 KBytes
[ 5] 9.00-10.00   sec  509 KBytes  4.17 Mbits/sec   72  5.66 KBytes
-----
[ ID] Interval      Transfer      Bitrate      Retr
[ 5] 0.00-10.00   sec  5.91 MBytes  4.96 Mbits/sec  760
[ 5] 0.00-10.00   sec  5.57 MBytes  4.68 Mbits/sec
sender
receiver

```

Fig. 8. Connection test between Host 1 and 6(after service created)

Once the service has been created, the `Iperf` tool shows that it is possible to communicate between users 1 and 6 with Internet access and performance limited to 4000 Mbits/sec.

The presented solution showed good performance when applied to small-dimension networks. However, when dealing with more extensive networks, as in industrial contexts, the problem becomes more challenging to solve, increasing the waiting time between the initiation of the user request and its installation. In this way, the next chapter will introduce

a new architecture so that the size of the network is not a problem for the installation of intents in the network.

3.4. Enabling E2E Network Orchestration in VINIA

Upon evaluation of the requirements for 6G networks [6] concerning the use of IBN, and considering the shortcomings of the architecture presented in Figure 4, we now propose a new architecture that enables End-to-End Network Orchestration in VINIA, in Figure 9. The architecture considers the latest developments in Network Automation & Orchestration and incorporates new insights and conclusions from the current state of the art. This architecture aims to provide a comprehensive and up-to-date framework offering significant improvements in efficiency and performance when dealing with Industrial networks. Moreover, it serves as a starting point for further research and development, and we believe it has the potential to make significant contributions to the field.

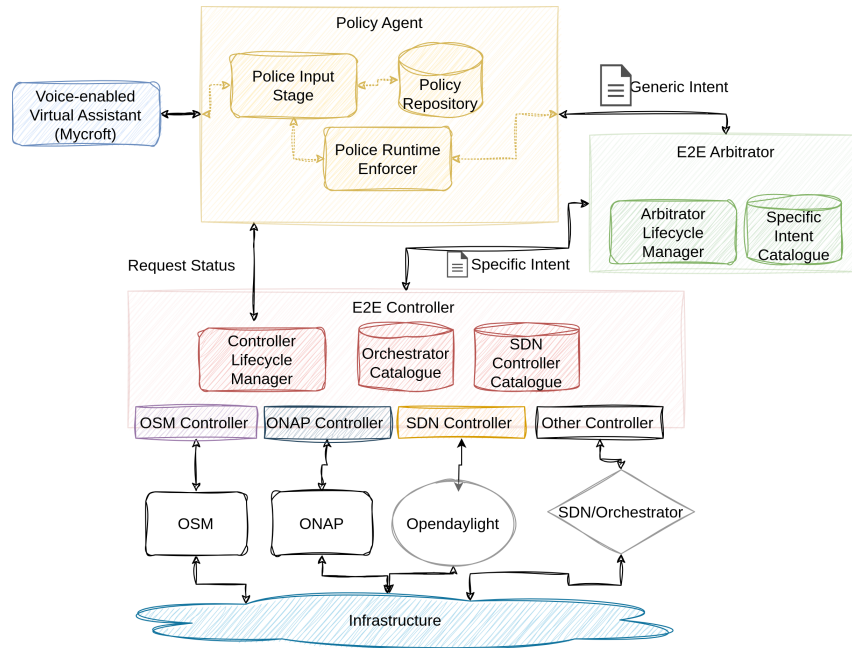


Fig. 9. New proposed architecture for VINIA

Regarding the user request capture and translation to an intent that involves the VVA and the Policy Agent, the architecture is unchanged from the one presented in Figure 4. The innovation comes when the compilation and installation process is initiated, where the Police Runtime Enforcer component is decomposed into several parts: E2E Arbitrator and E2E Controller. E2E Arbitrator is responsible for mediating between the high-level intents defined by network administrators and the underlying network infrastructure, ensuring that the intents are accurately translated. The E2E Arbitrator bridges the intent and

network management layers, ensuring that the supporting Industrial network infrastructure is correctly configured and the desired network behavior is achieved. It receives a generic intent, as presented in Listing 1.1, then interprets the request and assumes what configurations are necessary to implement it using the Catalogue, which contains all the network information and what NOs are available and capable of implementing the intent. The output from this Arbitrator is a specific Intent with details on the components involved and which technologies to use to implement it. The E2E Controller is the intermediary between the E2E Arbitrator and the NOs, automating the translation of network intents into network configurations. It has several key components to help it manage the network infrastructure: Controller Lifecycle Manager, Orchestrators Catalogues, SDN Catalogue, and Orchestrators Controllers. The Controller Lifecycle Manager is responsible for managing the lifecycle of the E2E Controller, including initializing, updating, and terminating the controller as needed. The Orchestrator Catalogue is a repository of information about available NOs, including their capabilities, functionalities, and dependencies. As Orchestrator Catalogue, SDN Controller Catalogue is a similar repository for SDN controllers, providing information about available controllers and their capabilities. The last component, Orchestrators Controller, starts the installation process, sending the network configurations to the respective orchestrator. When the E2E Controller receives the Specific Intent, it uses the information stored in the Orchestrator Catalogue or SDN Catalogue (depending on the associated technology in the Arbitrator) to compile the intent in network configurations. Once the intent is compiled in network configurations readable by the respective orchestrator, the E2E Controller sends those configurations to the orchestrator to implement the intent by configuring the underlying network resources. This enables the Industrial network infrastructure to be managed and automated flexibly and efficiently, ensuring that the network behaves in the desired way.

4. Conclusion and Future Directions

Using high-level network intents to define desired network behavior, IBN enables network administrators to manage heterogeneous network infrastructures, including industrial ones, more efficiently and agilely. However, implementing IBN within traditional SDN contexts is challenging. Some issues are limited scalability, integration with other network components, and difficulty translating high-level intents. In VINIA's new architecture, we have explored various IBN implementation strategies, with a special focus on leveraging NOs like ONAP or OSM to enhance effectiveness and scalability.

The proposed VINIA's architecture towards supporting different components (including proprietary ones) allows for defining an E2E network. VINIA ensures support for Network Slices and compatibility with heterogeneous network infrastructure configurations in Industrial Networks (e.g., 5G and B5G Networks). Following VINIA's core definitions, automating the networks allows for the redefinition of the network operating cost. Moreover, as requirements evolve or expand, VINIA can integrate additional components, minimizing disruptions and ensuring sustained performance even as the network grows.

Moving forward, we will focus on validating the introduced architecture in the 6G Brains project, particularly through slice use cases [10] demonstrating its effectiveness in real-world scenarios. We aim to test the new components for each orchestrator type to ensure seamless integration with existing technologies and infrastructure. Through the

use of orchestration in cooperation with monitoring information on the ongoing status of the network at the control plane, it is possible to have a disaggregated view of the network. The ability to control the different network segments of a E2E network allows for defining more complex intents to manage the different network's resources. The use of monitoring mechanisms as well as topology discovery mechanisms can be used to actively preclude QoS degradation. Towards this goal, in [10], a mechanism for enriching ONAP with information about the network was presented. Traditional SDN-based implementations are unaware of the E2E network. Therefore, using orchestrators such as OSM/ONAP improves the response to network changes, ensuring the E2E network can reconfigure quickly. Therefore, using orchestrators helps improve the dynamicity of the solution. Having a complete picture of the E2E network will allow for better monitoring, promoting the capability to detect points of failure/congestion. In the future, we aim to leverage the use of closed-control loops to detect problems in components managed by the VINIA system. In the scope of 6G BRAINS, we have been developing the first iteration of this concept, focused on the RAN. This concept relies on the metrics collected in the RAN for changing the type of network service provided in a specific RAN network slice.

Beyond network orchestration, we plan to expand the functionalities of the VVA as well as testing another VVA, Neon AI ¹¹ as an alternative to the use of Mycroft. Furthermore, we plan to enhance the language understanding capabilities of VVAs, making them more inclusive by supporting multiple languages, accents, dialects, and colloquialisms, testing alternatives to Mozilla's DeepSpeech ¹² Advances in natural language processing and machine learning techniques hold the potential to significantly improve VVA performance and accessibility for a broader range of users.

By addressing these aspects, we expect to overcome the constraints of conventional SDN-based IBN implementations, paving the way for a more efficient and integrated network management system. The continual evolution of IBN and VVA technologies will contribute to the progressive development and practical deployment of advanced network automation solutions in the industrial context.

Acknowledgments. This work was funded by the European Commission Horizon 2020 5G-PPP Program under Grant Agreement N. H2020-ICT-2020-2/101017226 "6G BRAINS: Bringing Reinforcement learning Into Radio Light Network for Massive Connections".

References

1. Kashif Mehmood, H. V. Kalpanie Mendis, Katina Kravetska, and Poul E. Heegaard. Intent-based network management and orchestration for smart distribution grids, 2021.
2. Talha Ahmed Khan and et al. Intent-based orchestration of network slices and resource assurance using machine learning. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–2, 2020.
3. Hong Chen, Ling Li, and Yong Chen. Explore success factors that impact artificial intelligence adoption on telecom industry in china. *Journal of Management Analytics*, 8(1):36–68, 2021.

¹¹ <https://neon.ai/>

¹² <https://github.com/mozilla/DeepSpeech>, which are used by both Mycroft AI and Neon AI.

4. Mounir Bensalem, Jasenka Dizdarević, Francisco Carpio, and Admela Jukan. The role of intent-based networking in ict supply chains. In *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, pages 1–6, 2021.
5. Raul Barbosa and Marco Araujo. Ai-driven human-centric control interfaces for industry 4.0 with role-based access. In *2022 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6, 2022.
6. Ömer Bulakçı (ed.) and et al. *Towards Sustainable and Trustworthy 6G: Challenges, Enablers, and Architectural Design*. Boston-Delft: now publishers, 2023.
7. Abdallah Moubayed, Abdallah Shami, and Anwer Al-Dulaimi. On end-to-end intelligent automation of 6g networks. *Future Internet*, 14(6), 2022.
8. Hatim Chergui, Adlen Ksentini, Luis Blanco, and Christos Verikoukis. Toward zero-touch management and orchestration of massive deployment of network slices in 6g. *IEEE Wireless Communications*, 29(1):86–93, 2022.
9. Karima Velasquez, David Perez Abreu, Marilia Curado, and Edmundo Monteiro. Resource orchestration in 5g and beyond: Challenges and opportunities. *Computer Communications*, 192:311–315, 2022.
10. João Fonseca and et al. 6g brains topology-aware industry-grade network slice management and orchestration. In *2023 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pages 418–423, 2023.
11. Masoud Narouei and Hassan Takabi. Automatic top-down role engineering framework using natural language processing techniques. In Raja Naeem Akram and Sushil Jajodia, editors, *Information Security Theory and Practice*. Springer International Publishing, 2015.
12. Aodi Liu, Xuehui Du, and Na Wang. Efficient access control permission decision engine based on machine learning. *Secur. Commun. Networks*, 2021:3970485:1–3970485:11, 2021.
13. Fan Deng and Li-Yong Zhang. Elimination of policy conflict to improve the pdp evaluation performance. *Journal of Network and Computer Applications*, 80:45–57, 2017.
14. Said Marouf, Mohamed Shehab, Anna Squicciarini, and Smitha Sundareswaran. Adaptive reordering and clustering-based framework for efficient xacml policy evaluation. *IEEE Transactions on Services Computing*, 4(4):300–313, 2011.
15. Santiago Pina Ros, Mario Lischka, and Félix Gómez Mármol. Graph-based xacml evaluation. In *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies, SACMAT '12*, page 83–92, New York, NY, USA, 2012. Association for Computing Machinery.
16. Manar Alohal, Daniel Takabi, and Eduardo Blanco. Automated extraction of attributes from natural language attribute-based access control (abac) policies. *Cybersecurity*, 2, 12 2019.
17. Eni. Gr eni 003 - v1.1.1 - experiential networked intelligence (eni); context-aware policy management gap analysis. Technical report, ETSI, 2018.
18. Yiming Wei, Mugen Peng, and Yaqiong Liu. Intent-based networks for 6g: Insights and challenges. *Digital Communications and Networks*, 6(3):270–280, 2020.
19. Engin Zeydan and Yekta Turk. Recent advances in intent-based networking: A survey. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–5, 2020.
20. Min Xie, Pedro Henrique Gomes, Jörg Niemöller, and Jens Patrick Waldemar. Intent-driven management for multi-vertical end-to-end network slicing services. In *2022 IEEE Globecom Workshops (GC Wkshps)*, pages 1285–1291, Dec 2022.
21. Vijay Varadharajan, Kallol Karmakar, Uday Tupakula, and Michael Hitchens. A policy-based security architecture for software-defined networks. *IEEE Transactions on Information Forensics and Security*, 14(4):897–912, 2019.
22. Cristian Cleder Machado and et al. Policy authoring for software-defined networking management. *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, IM 2015*, pages 216–224, 2015.
23. Douglas Comer and Adib Rastegarnia. OSDF: A framework for software defined network programming. *CCNC 2018 - 2018 15th IEEE Annual Consumer Communications and Networking Conference*, 2018-Janua(October 2017):1–4, 2018.

24. Francesca Moscatelli and et al. Demo: Service-driven management of vertical services in heterogeneous network slices. In *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pages 1–3, 2021.
25. Khizar Abbas and et al. Ensemble learning-based network data analytics for network slice orchestration and management: An intent-based networking mechanism. In *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pages 1–5, 2022.
26. Talha Ahmed Khan, Khizar Abbas, Afaq Muhammad, and Wang-Cheol Song. An intent-driven closed-loop platform for 5g network service orchestration. *Computers, Materials & Continua*, 70(3):4323–4340, 2022.
27. Flávio Meneses, Manuel Fernandes, Daniel Corujo, and Rui L. Aguiar. Slimano: An expandable framework for the management and orchestration of end-to-end network slices. In *2019 IEEE 8th International Conference on Cloud Networking (CloudNet)*, pages 1–6, 2019.
28. Sumbal Malik, Manzoor Ahmed Khan, Adam, Hesham El-Sayed, Jalal Khan, and Obaid Ullah. Implanting intelligence in 5g mobile networks;a practical approach. *Electronics*, 11(23), 2022.
29. Serge Fdida, Nikos Makris, Thanasis Korakis, Raffaele Bruno, Andrea Passarella, Panayiotis Andreou, Bartosz Belter, Cédric Crettaz, Walid Dabbous, Yuri Demchenko, and Raymond Knopp. Slices, a scientific instrument for the networking community. *Computer Communications*, 193:189–203, 2022.
30. Peter Bailis and Kyle Kingsbury. The network is reliable: An informal survey of real-world communications failures. *Queue*, 12(7):20–32, jul 2014.
31. Nick Feamster and Hari Balakrishnan. Detecting BGP configuration faults with static analysis. In *2nd Symposium on Networked Systems Design & Implementation (NSDI 05)*, Boston, MA, May 2005. USENIX Association.
32. Maximilian Wilhelm. Bgp security and confirmation biases. online 08 Nov 2023.
33. 3GPP TSG. Telecommunication management; study on scenarios for intent driven management services for mobile networks. Technical Report 28.812, 3GPP, set 2022.
34. 3GPP TSG. Management and orchestration; provisioning. Technical Specification 28.531, 3GPP, jun 2021.
35. TM Forum. Autonomous networks scenario realizations. Introductory Guide 1230A, TM Forum, may 2021.
36. TM Forum. Autonomous networks industry standards. Introductory Guide 1230B, TM Forum, may 2021.
37. TM Forum. Autonomous networks reference architecture. Introductory Guide 1251, TM Forum, may 2021.
38. TM Forum. Study of telecom industry intent meta-modeling approaches. Introductory Guide 1259, TM Forum, may 2021.
39. TM Forum. Autonomous networks business requirements and framework. Introductory Guide 1218, TM Forum, dec 2022.
40. Adeel Rafiq, Asif Mehmood, and Wang-Cheol Song. Intent-based slicing between containers in sdn overlay network. *Journal of Communications*, pages 237–244, 01 2020.
41. Federica Paganelli, Francesca Paradiso, Monica Gherardelli, and Giulia Galletti. Network service description model for vnf orchestration leveraging intent-based sdn interfaces. In *2017 IEEE Conference on Network Softwarization (NetSoft)*, pages 1–5, 2017.
42. Walter Cerroni and et al. Intent-based management and orchestration of heterogeneous open-flow/iot sdn domains. In *2017 IEEE Conference on Network Softwarization (NetSoft)*, pages 1–9, 2017.
43. Yoshiharu Tsuzaki and Yasuo Okabe. Reactive configuration updating for intent-based networking. In *2017 International Conference on Information Networking (ICOIN)*, pages 97–102, 2017.

44. Arthur Selle Jacobs, Ricardo José Pfitscher, Ronaldo Alves Ferreira, and Lisandro Zambenedetti Granville. Refining network intents for self-driving networks. In *Proceedings of the Afternoon Workshop on Self-Driving Networks*, SelfDN 2018, page 15–21, New York, NY, USA, 2018. Association for Computing Machinery.
45. Fred Aklamanu and et al. Intent-based real-time 5g cloud service provisioning. In *2018 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6, 2018.
46. Rami Akrem Addad, Diego Leonel Cadette Dutra, Miloud Bagaa, Tarik Taleb, Hannu Flinck, and Mehdi Namane. Benchmarking the onos intent interfaces to ease 5g service management. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2018.
47. Adeel Rafiq, Asif Mehmood, Talha Ahmed Khan, Khizar Abbas, Muhammad Afaq, and Wang-Cheol Song. Intent-based end-to-end network service orchestration system for multi-platforms. *Sustainability*, 12(7), 2020.
48. Khizar Abbas, Muhammad Afaq, Talha Ahmed Khan, Adeel Rafiq, and Wang-Cheol Song. Slicing the core network and radio access network domains through intent-based networking for 5g networks. *Electronics*, 9(10), 2020.
49. Orken Mamyrbayev, Nurbapa Mekebayev, Mussa Turdalyuly, Nurzhamal Oshanova, Tolga Ihsan Medeni, and Aigerim Yessentay. Voice identification using classification algorithms. In Yang (Cindy) Yi, editor, *Intelligent System and Computing*, Rijeka, 2019. IntechOpen.
50. Meenu Yadav, Dr. Vinod Kumar Verma, Chandra Yadav, and Jitendra Verma. Mlpgi: Multilayer perceptron-based gender identification over voice samples in supervised machine learning. In *Applications of Machine Learning*, pages 353–364, 05 2020.
51. Mucahit Buyukyilmaz and Ali Osman Cibikdiken. Voice gender recognition using deep learning. In *Proceedings of 2016 International Conference on Modeling, Simulation and Optimization Technologies and Applications (MSOTA2016)*, pages 409–411. Atlantis Press, 2016/12.
52. Orken Mamyrbayev, Nurbapa Mekebayev, Mussa Turdalyuly, Nurzhamal Oshanova, Tolga Ihsan Medeni, and Aigerim Yessentay. Voice identification using classification algorithms. In Yang (Cindy) Yi, editor, *Intelligent System and Computing*, chapter 3. IntechOpen, Rijeka, 2019.
53. M Murugappan, Nurul Qasturi Idayu Baharuddin, and S Jerritta. Dwt and mfcc based human emotional speech classification using lda. In *2012 International Conference on Biomedical Engineering (ICoBE)*, pages 203–206, 2012.
54. Philip Loizou. NOIZEUS: A noisy speech corpus for evaluation of speech enhancement algorithm.
55. John S. Garofolo and et al. TIMIT Acoustic-Phonetic Continuous Speech Corpus.
56. Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2015-Augus:5206–5210, 2015.

Raul Barbosa is currently pursuing his PhD at the University of Aveiro and is affiliated with the Telecommunications Institute. His research centers around the development of AI-based systems to enhance networks in autonomous driving, industrial settings, and other vertical domains. Alongside his academic pursuits, Raul holds the role of Professional II-Connectivity & NW Engineer in Capgemini’s R&D Telecom team, actively contributing to various national and international research projects.

João Pedro Fonseca is a PhD Student enrolled in the MAPi Programme, and associated with the Telecommunications Institute. He currently pursues research on Vertical-aware and Dynamic System Orchestration. He is a Connectivity & NW Engineer at the R&D Telecom team of Capgemini, contributing to national and international research projects.

Marco Araújo is a telecommunications expert with over 15 years of experience working with telecom operators like NOS, T-Mobile, Vodafone, Oi, Vivo, and Telefonica. He holds a PhD from Stockholm University, specializing in applications of telecommunications systems engineering for the automotive and aerospace sectors. Leading R&D projects such as Horizon Europe and collaborating with the European Space Agency, Marco manages budgets as a senior manager in Portugal's R&D department, ensuring innovative progress.

Daniel Corujo is an Associate Professor at the University of Aveiro and a Researcher at the Telecommunications Institute, both in Portugal. He pursues research on programmatic mobile networking in national and international research projects, holding several research papers, standards and patents. He is Vice-chair of the IEEE COMSOC PT Chapter.

Received: February 13, 2023; Accepted: November 15, 2023.