ComSIS **Computer Science and Information Systems**

Volume 17, Number 3, October 2020

## Contents

**Special Section: Parallel Computing and Applications**

**Special Section: Edge Computing**

**Special Section: Advances in Databases and Information Systems**

ISSN: 1820-0214 (Print) 2406-1018 (Online)

Computer Science and Information Systems

Vol 17, No 3, October 2020

ComSIS

# Computer Science and Information Systems

Published by ComSIS Consortium

**Special Issue**

Volume 17, Number 3
October 2020

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
Faculty of Economics, Subotica, Serbia
Technical Faculty “Mihajlo Pupin”, Zrenjanin, Serbia

**University of Montenegro:**
Faculty of Economics, Podgorica, Montenegro

**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 17, Number 3, 2020**
**Novi Sad**

# Computer Science and Information Systems

Special issue: Advances in Databases, Information Systems, Parallel Computing and Edge Computing

**ISSN: 1820-0214 (Print) 2406-1018 (Online)**

The ComSIS journal is sponsored by:

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

Com SIS

# 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 2019 two-year impact factor 0.927,
- 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 (MS Word or 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 Word or 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 17, Number 3, Special Issue, October 2020

## CONTENTS

### Special Section: Parallel Computing and Applications

### Special Section: Edge Computing

## Special Section: Advances in Databases and Information Systems

Computer Science and Information Systems 17(3): i–ii https://doi.org/10.2298/CSIS200300iS

# Guest Editorial
# Parallel Computing and Applications

Hong Shen[1], Hui Tian[2], and Yingpeng Sang[1]

[1] Sun Yat-Sen University
China
{shenh3,sangyp}@mail.sysu.edu.cn
[2] Beijing Jiaotong University
China
htian@bjtu.edu.cn

## 1. Background

The research on parallel computing and relevant applications is undergoing deep changes and generating far-reaching impacts. Abundant theory, design and analysis on parallel and distributed computing systems have been provided in this field. The objective of this special session is to publish and overview the cutting edge research in parallel computing and applications. This special session includes papers based on the presentation at the 10th International Conference on Parallel Architectures, Algorithms and Programming (PAAP 2019), and the 20th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT 2019).

## 2. Scanning the session

The topics of this special session covers a new model to predict the age and number of criminal suspects through the feature modeling of historical data, an improved heuristic-dynamic programming algorithm for rectangular cutting problem, energy-efficient algorithms of non-linear k-barrier coverage in mobile sensor network, maximizing concurrent data flows in Multi-radio Multi-channel wireless mesh networks, and an improved Monte Carlo Localization Boxed algorithm for node localization in wireless sensor networks.

The first paper “A New Model for Predicting the Attributes of Suspects,” by Chuyue Zhang et al. proposes a new model to predict the age and number of suspects through the feature modeling of historical data. 9 machine learning algorithms and Deep Neural Networks are used to extract the numerical features. Convolutional Neural Networks and Long Short-Term Memory are used to extract the text features of case description. These two types of features are combined and fed into fully connected layer and softmax layer. The experimental results show that the new model improves accuracy by 3% in predicting the number of suspects and improves accuracy by 12% in predicting the number of suspects.

The second paper “An Improved Heuristic-Dynamic Programming Algorithm for Rectangular Cutting Problem,” by Aihua Yin et al. proposes an improved heuristic-dynamic programming algorithm for rectangular cutting problem. The objective is to cut some rectangles in a given shape and direction without overlapping the defects from the rectangular plate and maximize some profit associated. In this algorithm, the discrete set contains not only the solution of one-dimensional knapsack problem with small rectangular block width and height, but also the cutting positions of one unit outside four boundaries of each defect. The computational experimental results show that the computation time is less than that of the latest literature algorithms.

The third paper "Energy-Efficient Non-linear K-Barrier Coverage in Mobile Sensor Network," by Zijing Ma et al. proposes two algorithms to form non-linear k-barrier coverage energy-efficiently. The algorithms use the horizontal virtual force model by considering both the Euclidean distance and horizontal angle between two sensors. The first algorithm always chooses the mobile sensor chain with the largest horizontal virtual force and then moves it to construct the barrier, called mobile sensor chain movement algorithm. The other algorithm chooses the mobile sensor with the largest horizontal virtual force and moves it to construct the barrier, other than the mobile sensor chain, called single sensor movement algorithm. Simulation results show that the algorithms significantly reduce the movements of mobile sensors compared to a linear k-barrier coverage algorithm.

The fourth paper "Maximize Concurrent Data Flows in Multi-radio Multi-channel Wireless Mesh Networks," by Zhanmao Cao et al. analyzes traffic behaviors and designs a coexisting algorithm to maximize the number of concurrent data flows, in Multi-radio Multi-channel wireless mesh networks. Simulations are conducted in combinatorial cases of channel and radio with various traffic requests of multiple pairs. The experimental results show the efficacy of the coexisting algorithm over a randomly generated topology. This scheme can be used to develop routing and scheduling solutions for various multi-flow network applications through prior computing.

Aiming at the problem of low sampling efficiency and high demand for anchor node density of traditional Monte Carlo Localization Boxed algorithm, the fifth paper "An Improved MCB Localization Algorithm Based on Weighted RSSI and Motion Prediction," by Chunyue Zhou et al. proposes an improved algorithm based on historical anchor node information and the received signal strength indicator (RSSI) ranging weight which can effectively constrain sampling area of the node to be located. Moreover, the RSSI ranging of the surrounding anchors and the neighbor nodes is used to provide references for the position sampling weights of the nodes, and an improved motion model is proposed to further restrict the sampling area in direction. The simulation results show that the improved Monte Carlo Localization Boxed algorithm effectively improves the accuracy and efficiency of localization.

Computer Science and Information Systems 17(3): iii–iv https://doi.org/10.2298/CSIS200300iiiG

# Guest Editorial
# Edge Computing

Honghao Gao[1], Yuyu Yin[2], and Jung Yoon Kim[3]

[1] Shanghai University
China
[2] Hangzhou Dianzi University
China
[3] Gachon University
South Korea

Edge computing is proposed as a new computing paradigm where resources like computation and storage are placed closer to data and information sources. On one hand, edge computing helps the cloud to extend its services to the edge of the network, which improves the response time and user experience. On the other hand, the development of the Internet of Things (IoT) leads to a huge amount of data generated on the user side and there are also a lot of smart devices available, it is a natural way to process data on the edge. In a word, the edge computing paradigm greatly saves the bandwidth of the backbone network and improves the end-to-end latency. It brings new possibilities for complex applications, intelligent services, novel security, and privacy solutions, especially for the researches which depends on the huge amount of data. This special issue features seven selected papers with high quality.

The first article titled ”VDRF: Sensing the Defect Information to Risk Level of Vehicle Recall based on Bert Communication Model” proposes a risk level prediction method based on the language pre-training model Bert. The prediction method can transform the defect information into the risk level of the vehicle and then predict vehicle recall automatically. The second paper, ”A Homomorphic-encryption-based Vertical Federated Learning Scheme for Rick Management”, designs a vertical federated learning system for Bayesian machine learning with the homophobic encryption. Specifically, during the training progress, raw data are leaving locally, and encrypted model information is exchanged. The third paper is ”A Novel Data-Driven Intelligent Computing Method for the Secure Control of a Benchmark Microgrid System”, which investigates the secure control problem of a benchmark micro-grid with system uncertainties by using data-driven edge computing technology. In detail, it first formulates the state-space function of the benchmark micro-grid system and then derives a novel data-driven intelligent computing method. The fourth paper, ”A Recommendations Model with Multiaspect Awareness and Hierarchical User-Product Attention Mechanisms”, proposes a novel model to capture the varying attention of a user for different products by using a multilayer attention framework. Specifically, two individual hierarchical attention networks are used to encode the users and products to learn the user preferences and product characteristics from review texts. And an attention network is designed to reflect the adaptive change in the user preferences for each aspect of the targeted product in terms of the rating and review. The fifth paper is ”A Study on the Development of a Light Scattering Particulate Matter Sensor and Monitoring System”, and it develops a light scattering type PM sensor that can be manufactured at a low cost and can measure PM in real-time. Moreover, the paper builds a big data system that can systematically store and analyze the data collected through the developed sensor, as well as an environment where PM states can be monitored mobile using such data. The sixth paper titled ”A Load Balancing Scheme for Gaming Server applying Reinforcement Learning in IoT” proposes an agent that applies a deep reinforced learning method to distribute loads for gaming servers. The agent has accomplished this by measuring network loads and analyzing a large amount of user data. The seventh paper is ”Graph Embedding Code Prediction Model Integrating Semantic Feature”, which proposes a graph embedding model that integrates semantic features. The model extracts the structural paths between the nodes in the source code file’s Abstract Syntax Tree (AST) and converts them into a training graph.

**Acknowledgments.** The guest editors are thankful to our reviewers for their effort in reviewing the manuscripts. We also thank the Editor-in-Chief, Dr. Mirjana Ivanovic, for her supportive guidance during the entire process.

Computer Science and Information Systems 17(3): v–vi https://doi.org/10.2298/CSIS200300vG

# Guest Editorial Advances in Databases and Information Systems

Johann Gamper[1], Mirjana Ivanović[2], and Robert Wrembel[3]

[1] Free University of Bozen-Bolzano
Italy
[2] University of Novi Sad
Serbia
[3] Poznan University of Technolgoy
Poland

This special section includes the best papers from the workshops co-located with the 23rd European Conference on Advances in Databases and Information Systems - ADBIS (http://adbis.eu), which was held in Slovenia in 2019. From the top 10 papers invited to this special issue, 3 were accepted and are included in this volume. These papers report extended research w.r.t. the original ADBIS Workshop papers. They cover the following tree topics: (1) process discovery, (2) crowd counting , and (3) user preference modeling.

The first paper entitled *Improving the Performance of Process Discovery Algorithms by Instance Selection* is contributed by Mohammadreza Fani Sani, Sebastiaan J. van Zelst, and Wil van der Aalst. The paper addresses the problem of increasing performance of a process discovery algorithm. Such an algorithm automatically discovers process models by analyzing events generated during the execution of the algorithm. The events are stored in a log. Process discovery is a complex, time consuming task, thus reducing its execution time is one of the challenges. One approach to reducing the execution time is to reduce the volume of events that need to be analyzed by the algorithm. In this paper the authors analyzed 7 alternative biased sampling methods and experimentally assessed their impact on increasing the algorithm scalability on various event logs.

The second paper entitled *Crowd counting a la Bourdieu, Automated estimation of the number of people*, is contributed by Karolina Przybylek and Illia Shkroba. Recently the problem of crowd counting is emergent in different situations. Crowd counting is important in public safety (like crushing between people, risk of spreading infectious disease), politics (like protest organisation) journalism (like accuracy of the estimation of the ground truth supporting an article) and other areas. The paper investigated models for crowd counting that are inspired by the observations of famous sociologist Pierre Bourdieu. Authors in their experiments achieved very competitive result suitable for low computational power and energy efficient architectures.

The third paper entitled *Visual E-Commerce Values Filtering Framework with Spatial Database Metric* is contributed by Michal Kopecky and Peter Vojtáš. Recommender systems have been subject to intensive studies in the past decade, in particular in the context of e-commerce. A core component of such systems is the modeling of user preferences. This paper starts from the Fagin-Lotem-Naor model for preference modeling and combines it with the challenge-response framework for the translation of models into programs and, to make the models more intuitive, with spatial database features. This allows on one hand to visualize the models and on the other hand to define new metrics.

Computer Science and Information Systems 17(3):705–715 https://doi.org/10.2298/CSIS200107016Z

# A New Model for Predicting the Attributes of Suspects

Chuyue Zhang[1], Xiaofan Zhao[1*], Manchun Cai[1], Dawei Wang[2] and
Luzhe Cao[1]

[1] School of Information Technology and Cyber Security
People’s Public Security University of China
Beijing, China
zhaoxiaofan@ppsuc.edu.cn
[2] College of Criminology
People’s Public Security University of China
Beijing, China
649661694@qq.com

**Abstract.** In this paper, we propose a new model to predict the age and number of suspects through the feature modeling of historical data. We discrete the case information into values of 20 dimensions. After feature selection, we use 9 machine learning algorithms and Deep Neural Networks to extract the numerical features. In addition, we use Convolutional Neural Networks and Long Short-Term Memory to extract the text features of case description. These two types of features are fused and fed into fully connected layer and softmax layer. This work is an extension of our short conference proceeding paper. The experimental results show that the new model improved accuracy by 3% in predicting the number of suspects and improved accuracy by 12% in predicting the number of suspects. To the best of our knowledge, it is the first time to combine machine learning and deep learning in crime prediction.

**Keywords:** crime prediction, suspect prediction, machine learning, deep learning.

## 1. Introduction

Criminal activities show certain distribution characteristics in time and space, which indicates that criminal activities are not completely random. The offenders often carry out the crime selectively according to the time and place. It is a hot topic to discover the law of crime from criminal activities.

With the development of statistics and criminology, the cognition and analysis of criminal behavior are gradually improved. Big data has many applications in the field of public security and criminology. Crime prediction provides assistance for crime prevention, public security prevention and control, case detection and police decision-making and has become a hot research topic nowadays.

In this paper, deep learning algorithms and machine learning algorithms are applied to the prediction of criminal suspects. Through a large number of historical data, the relationship between the information of the case and the attributes of the suspect is mined. Compared with the traditional person-oriented case investigation, the model based on big data can provide police officers with more objective and comprehensive auxiliary information and help them find the perpetrators as soon as possible. The effective and accurate information of the suspect's characteristics is of great significance for the rapid detection of the case. This can not only reduce the police useless work, but also help the victims to recover the loss. For the construction of a harmonious society, it has great significance.

The main work of this paper is to use the historical data of the theft crime to predict the age of the suspect in the theft case through deep learning. Based on the idea of ensemble learning, this paper combines deep learning with machine learning, and uses textual data and numerical data to predict the age of suspects. When a new crime occurs, it can provide the predicted results of age and number.

## 2. Related Works

Police have embraced predictive analytics and data-driven metrics to improve law enforcement tactics, practice, and strategy [1]. Predictive policing [2] methods fall into four general categories: predicting the occurrence of crime, predicting the suspects, predicting the possibility of recommitting the crime and predicting the victims of crimes. This paper focuses on predicting the suspects.

The information analysis of the traditional case solving mainly depends on the experience of the investigators. Crime has the characteristics of time and space aggregation [3]. Melo S N et al. [4] analyzed that crime has certain regularity and stability over a long period of time. Sagovsky et al. [5] analyzed the relationship between season, temperature and property crimes. Almanie et al. [6] analyzed the economic factors, demographic factors for crime and used Decision Tree classifier and Naive Bayesian classifier to predict crime types. Michael Oyinloye et al. [7] analyzed the influence of age, income and education on criminal behavior. Song et al. [8] analyzed the impact of different border areas on different types of crime. For the suspect's prediction, TOLLENAAR N et al [9] use statistical method to predicate general recidivism, violent recidivism and sexual recidivism. Based on the case information and victim information, LI Ronggang et al. [10] use Support Vector Machine algorithm to predict the suspect's gender, age, race, etc. Based on date and location, crime type, criminal ID and the acquaintances, Vural MS et al. [11] use Naive Bayesian Model to predict criminal of particular crime incident. Based on the features of criminals in criminal case, SUN Feifei et al. [12] uses random forest model to predict possible suspects. But some of the input data in these studies is only known after solving the case, such as suspect age, criminal history, acquaintances, etc. In the real situation, we only know the objective information of the case, and we could predict the characteristics of suspects from it. In this paper, we only use case information obtained by investigators to predict the number of suspects.

The object of this paper is the crime of theft. We extract case features such as the time of the case, loss amount, method, places and so on, and turn them into numerical data according to certain rules. In order to make up for the feature loss caused by data dispersion, we added more comprehensive text information on case description. In addition, we combine machine learning and deep learning in training numerical data. For text data, we use Convolutional Neural Network (CNN) to capture the local features and Long Short-Term Memory (LSTM) to capture text sequence features. When a new case occurs, the model can provide investigators with the prediction of age of suspects.

## 3. Data Preprocessing

In this paper, the time, place, loss amount, methods and case categories are preprocessed and labeled as shown in tables 1 to 6.

**Table 1.** The data processing rules of case categories.

| Label | Categories | Number |
|---|---|---|
| 1 | pickpocketing | 5500 |
| 2 | theft of property in the car | 4872 |
| 3 | household theft | 5500 |
| 4 | theft of non-motor vehicles | 5500 |

**Table 2.** The data processing rules of time period.

| Label | Time (24 hours) | Note |
|---|---|---|
| 1 | 0:00~8:00 | Including 8:00 |
| 2 | 8:00~12:00 | Including 12:00 |
| 3 | 12:00~14:00 | Including 14:00 |
| 4 | 14:00~18:00 | Including 18:00 |
| 5 | 18:00~22:00 | Including 22:00 |
| 6 | 22:00~24:00 | Including 24:00 |

**Table 3.** The data processing rules of places.

| Label | Categories | Instructions |
|---|---|---|
| 1 | Residential | Residence, dormitory |
| 2 | Traffic area | Subway station, bus, roadside, expressway, etc. |
| 3 | Office area | Schools, hospitals, parking lots, offices, etc. |
| 4 | Entertainment area | Shopping malls, vegetable markets, street shops, catering places, Internet cafes and so on. |

**Table 4.** The data processing rules of methods.

| Label | Categories | Instructions |
|---|---|---|
| 1 | Others | Stolen goods from trucks, Unarmed climbing, Decoy |
| 2 | Pickpocketing | Pickpocketing, By the way to steal |
| 3 | Technical unlock | Insert card unlock, Poke the lock, Tin foil unlock, Technical unlock |
| 4 | From the window to enter | Break Windows, Break glass |
| 5 | From the roof to enter | From the roof, From the vent |
| 6 | From the wall to enter | Climb over the wall, Break walls |
| 7 | From the door to enter (Door damage) | Expand seam, Destroy anti-theft net (column), Destroy the door body |
| 8 | From the door to enter (No damage to the door) | Deceive into the door, Follow others into the door |
| 9 | Violence unlocked | Break the lock core, Pliers cut the lock, Break the lock |
| 10 | Violence hit the car | Smashing car windows, Pry the trunk of the car |
| 11 | Theft through the window | Get the key through the window with a pole, Open the window and steal |
| 12 | Stolen vehicle | Towing, dragged, Traction |

**Table 5.** The data processing rules of loss amount.

| Label | money (RMB) | Note |
|---|---|---|
| 1 | 0~900 | Including 900 |
| 2 | 900~1880 | Including 1880 |
| 3 | 1880~3600 | Including 3600 |
| 4 | 3600~1000000 | More than 3600 |

**Table 6.** The data processing rules of weather.

| Weather | sunny | Rain or snow | Others |
|---|---|---|---|
| Daytime weather | 1 | 2 | 3 |
| Night weather | 1 | 2 | 3 |

The age predicted in this article is the suspect's age at the time of the crime. The prediction results are designed as a four-category problem. Considering the equilibrium of the data distribution, we take 1/4, median and 3/4 of the loss amount as the dividing point and divide the loss amount ages into four categories. The first quartile is 25. The median is 32. The third quartile is 42.

The number of suspects in this data set is extremely unbalanced. In order to balance the data, this paper unifies the label of more than or equal to two people committing the crime as 2, and the label of single person committing the crime as 1.

According to different time scales, the time information extracted includes year, quarter, month, ten days, day, week and time period. The location is described by latitude, longitude, administrative districts and place types. The day temperature, night temperature, day wind, night wind, day weather and night weather are also added to this model as auxiliary information. Example of pre-processed data are shown in table 7. Please refer to the conference paper for detailed description.

**Table 7.** Example of pre-processed data.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| inputs | numerical data | year | 2018 | category | 1 | week | 5 | night weather | 1 |
| | | month | 5 | method | 9 | time period | 5 | day temperature | 10 |
| | | day | 20 | longtitude | 11.11 | loss amount | 3 | night temperature | 0 |
| | | quarter | 2 | latitude | 11.11 | districts | 2 | day wind | 3 |
| | | ten-days | 3 | day weather | 1 | places | 3 | night wind | 3 |
| | text data | On May 20, 2018, 20, alarm, * * * id number * * * * *, tel * * * *, address * * * *, the alarm person lost * * * in * * * *, The lost items were valued at * * *. | | | | | | | |
| outputs | age | 2 | | | | | | | |
| | number | 1 | | | | | | | |

## 4. Model

We remove the feature with the lowest contribution by mutual information and Chi-square test, in order to reduce the influence of irrelevant factors on the prediction results.

### 4.1. Numerical Data

Different machine learning algorithms have different advantages in data processing. Using the idea of ensemble learning, we build a group of classifiers composed of Logistics Regression (LR), Support Vector Machine (SVM), Naive Bayesian (NB), XGBoost, k-NearestNeighbor (KNN), GradientTree, Boosting (GBDT), adaboost, Quadratic Discriminant Analysis (QDA) and Linear Discriminant Analysis(LDA), and input their output results and raw data into the neural network. Although some of the above machine learning algorithms have used the idea of ensemble learning, in this paper, the results of the above nine algorithms are used as the input of the neural network, and the weight distribution of each input is completed by the neural network. The processing of numerical data is shown in figure 1.

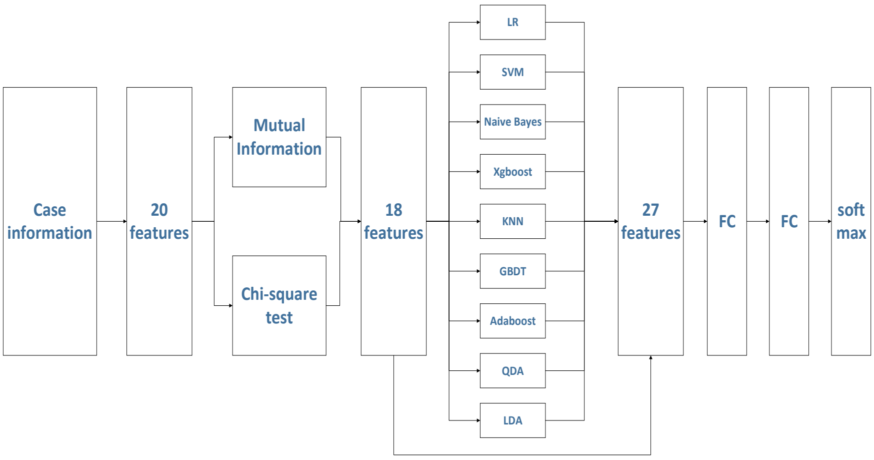


**Fig. 1.** Model flow for numerical data

Fully connected (FC) layer is to make linearly indivisible data become linearly separable. Formula 1 is as follows. Softmax is to make the features more distinct. Formula 2 is as follows.

$$h_{W,b}(x) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} (\sum_{i=1}^{n} W_i x_i + b) \qquad (1)$$

$$y_i = S(h)_i = \frac{e^{hi}}{\sum_{j=1}^{n} e^{hj}}, \qquad i = 1, \dots, n \qquad (2)$$

In formula 1, $W_i$ is the weight of the neuron. $b$ is the bias of the neuron. $n$ is the number of neurons. In formula 2, $h$ is the input and $y_i$ is the probability of being in the n$_{th}$ class.

Considering overfitting, we did not use Decision Tree (DT) and Random Forest Classifier (RFC). In the prediction of age, they can achieve 91% accuracy on the training set, but only 51% accuracy on the test set. The machine learning experimental results are shown in table 8. If the machine learning algorithm is overfitting, the data classification selected as the training set will be very good, but if a new case occurs, the classification result would be not good.

**Table 8.** Machine learning experimental results for the prediction of age.

| Algorithm | Accuracy on the training set | Accuracy on the test set |
|---|---|---|
| LR | 0.32 | 0.31 |
| SVM | 0.50 | 0.41 |
| bayes | 0.32 | 0.33 |
| xgboost | 0.32 | 0.32 |
| knn | 0.43 | 0.39 |

| | | |
|---|---|---|
| GBDT | 0.36 | 0.35 |
| Adaboost | 0.47 | 0.45 |
| LDA | 0.32 | 0.33 |
| QDA | 0.36 | 0.35 |
| RFC | 0.91 | 0.51 |
| DT | 0.91 | 0.51 |

### 4.2. Text Data

The time, place, loss amount, methods and case categories of the case are all from the structured tables in the public security system. But these don't fully describe the case, such as the name of the lost item, the details of the crime scene description. Therefore, we added the case description information in the form of text.

The case description is filled in by the police officer handling the case, ranging from 5 to 150 words. This paper uses CNN and LSTM for text processing. CNN's local perception and weight sharing mechanism enables the algorithm to capture local features. The gate mechanism of LSTM selectively letting information through enables the algorithm to learn long distance dependencies. The processing of text data is shown in figure 2.

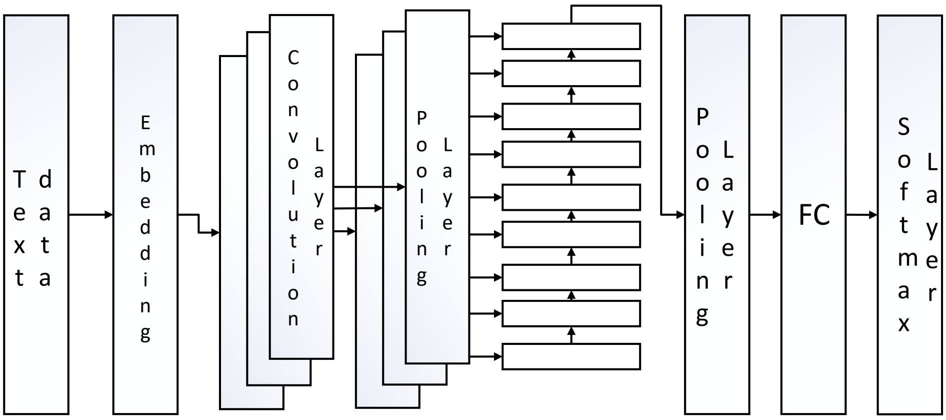


**Fig. 2.** Text data model structure

### 4.3. Model construction

In this paper, we use a parametric-matrix-based method to fuse the two types of features. The last layer of both numerical data model and text data model is softmax layer with the output shape of 4. The fusion formula is as follows.

$$X_f = W_n \circ X_n + W_t \circ X_t \qquad (3)$$

$\circ$ in formula is Hadamard product, $X_n$ and $X_t$ are the processed numerical and textual features. $W_n$ and $W_t$ are the learnable parameters which is used to adjust the degrees.

After fusion, we put them into the full connection layer. Finally, classification tasks are completed through softmax layer. The model structure is shown in figure 3.

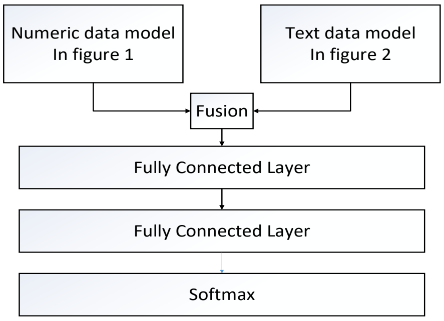


**Fig. 3.** Model structure

# 5. Experiments

## 5.1. Experimental Data

The data sets in this paper are all real data from the public security system, totaling 21372 pieces. We judge the practicability of this model by predicting the number and the age of suspects, named as test set 1 and test set 2 respectively.

Training sets, validation sets, and test sets are allocated according to the ratio of 7:2:1. The optimal model is determined by training set and cross validation set. The precision rate, recall rate, and F value of the test set are calculated as model evaluation.

## 5.2. Comparative Experiments

The comparative experiments are as follows.

- □DNN (Use 20 - dimensional numerical data)
- □Numerical -only model (Figure 1 model)
- Text-only model (Figure 2 model)
- □Model in the conference paper

The experimental results for data 1 are shown in table 9. The experimental results for data 2 are shown in table 10.

**Table 9.** Experimental results for predicting the number of suspects.

| Algorithm | Precision rate | Recall rate | F-values |
|---|---|---|---|
| DNN | 71 | 73 | 72 |
| numerical -only model | 90 | 90 | 90 |
| text-only model | 65 | 64 | 64 |
| Model in the conference paper | 91 | 91 | 91 |
| The new model | 94 | 93 | 93 |

**Table 9.** Experimental results for predicting the age of suspects.

| Algorithm | Precision rate | Recall rate | F-values |
|---|---|---|---|
| DNN | 38 | 37 | 37 |
| numerical -only model | 57 | 56 | 56 |
| text-only model | 43 | 43 | 43 |
| Model in the conference paper | 53 | 51 | 52 |
| The new model | 65 | 65 | 65 |

### 5.3. Results Analysis

The experimental results are analyzed as follows:

1) This paper proves the possibility of predicting the age and number of suspects from the historical data of the case. The numerical discretization rules we define by ourselves are reasonable and feasible.

2) Compared to the conference model, the improvement of this model is the combination of machine learning and deep learning in numerical data processing. In addition, the LSTM is added. Experimental results show that these two improvements can improve accuracy by 3% in predicting the number of suspects and improved accuracy by 12% in predicting the number of suspects.

3) This model is not very good at predicting the age of suspects. When a crime is committed by more than one person, this model can only recommend the most likely age interval. This simulation is not applicable to older suspects who commit crimes together with younger suspects.

## 6. Conclusion

The big data algorithm is used to model the historical data and guide the public security decision-making. In this paper, the age of suspects and the number of suspects are predicted by modeling the features of real case information. Different machine learning algorithms have different advantages. We combine them, assign weights through the neural network. In addition, we add text data in order to make more accurate predictions.

When new cases occur, our model can provide the police with decision support. The model achieves 94 percent accurate in predicting the number of suspects and 65 percent accurate in predicting the age of suspects. Our model has strong practical significance. How to improve the ability of predicting the suspect's age is the next step.

**Acknowledgment** This work was supported by National Key R&D Program Project (Grant No.2018YFC0809802), the Fundamental Research Funds for the Central Universities (Grant No.2018JKF609), Specialized Research Fund of Higher Education of China (Grant No. 2019ssky012).

## References

1. Ferguson, Andrew Guthrie, Policing Predictive Policing. Washington University Law Review, Vol. 94, No. 5, 2017.
2. Perry, Walter L., Brian McInnis, Carter C. Price, Susan Smith, and John S. Hollywood, Predictive Policing: The Role of Crime Forecasting in Law Enforcement Operations. Santa Monica, CA: RAND Corporation, 2013.
3. Grubesic T H , Mack E A . Spatio-Temporal Interaction of Urban Crime[J]. journal of quantitative criminology, 2008, 24(3):285-306.
4. Melo S N, Matias L F, Andresen M A. Crime concentrations and similarities in spatial crime patterns in a Brazilian context[J]. Applied Geography, 2015, 62:314-324.
5. Sagovsky A , Johnson S D . When Does Repeat Burglary Victimisation Occur[J]. Australian and New Zealand Journal of Criminology, 2007, 40(1):1-26.
6. Almanie T , Mirza R , Lor E . Crime Prediction Based On Crime Types And Using Spatial And Temporal Criminal Hotspots[J]. Computer Science, 2015.
7. Michael, Oyinloye , Olamiju, Isaac , Otokiti, Kolade. Spatial Distribution of Crime in Akure, Nigeria: The GIS Perspectives SCIREA Journal of Geosciences. Journal of Geosciences. 2017,2. 21-38.
8. Song J , Andresen M A , Brantingham P L , et al. Crime on the edges: patterns of crime and land use change[J]. Cartography and geographic information science, 2017, 44(1):51-61.
9. TOLLENAAR N,HEIJDEN P G M V D. Which method predicts recidivism best: a comparison of statistical, machine learning and data mining predictive models [J]. Journal of the Royal Statistical Society,2013,176(2):565-584.
10. LI Ronggang, SUN Chunhua, JI Jianrui. Suspect characteristics prediction based on support vector machine[J]. Computer Engineering. 2017,43(11): 198-203.
11. Vural M S , G?K M . Criminal prediction using Naive Bayes theory[J]. Neural Computing and Applications, 2016:1-12.
12. SUN Feifei,CAO Zhuo,XIAO Xiaolei. Application of an improved random forest based classifier in crime prediction domain[J]. Journal of Intelligence,2014,33(10): 148-152.Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Databases. Morgan Kaufmann, Santiago, Chile, 487-499. (1994)

**Chuyue Zhang** is a postgraduate student in People’s Public Security University of China. Her main research interest is deep learning and crime prediction. E-mail: 649661694@qq.com

**Xiaofan Zhao** (corresponding author) is a PhD and teacher. Her research interest is data mining. She is the corresponding author of this paper. Email: zhaoxiaofan@ppsuc.edu.cn

**Manchun Cai** is a PhD and associate professor. His research interest is cryptology. E-mail: caimanchun@ppsuc.edu.cn

**Dawei Wang** is a PhD and professor. His research interest is criminology. E-mail: wdw_ppsuc@163.com.

**Luzhe Cao** is a postgraduate student in People’s Public Security University of China. His research interest is security situation awareness. E-mail: 867987100@qq.com

*Received: January 01, 2020; Accepted: June 18, 2020*

Computer Science and Information Systems 17(3):717–735 https://doi.org/10.2298/CSIS200125017Y

# An Improved Heuristic-Dynamic Programming Algorithm for Rectangular Cutting Problem

Aihua Yin, Chong Chen, Dongping Hu, Jianghai Huang and Fan Yang

School of Software and Internet of Things Engineering, Jiangxi University of Finance and Economics, Nanchang Jiangxi, China.
Dongping_hu337@jxufe.edu.cn

**Abstract.** In this paper, the two-dimensional cutting problem with defects is discussed. The objective is to cut some rectangles in a given shape and direction without overlapping the defects from the rectangular plate and maximize some profit associated. An Improved Heuristic-Dynamic Program (IHDP) is presented to solve the problem. In this algorithm, the discrete set contains not only the solution of one-dimensional knapsack problem with small rectangular block width and height, but also the cutting positions of one unit outside four boundaries of each defect. In addition, the denormalization recursive method is used to further decompose the sub problem with defects. The algorithm computes thousands of typical instances. The computational experimental results show that IHDP obtains most of the optimal solution of these instances, and its computation time is less than that of the latest literature algorithms.

**Keywords:** Guillotine, Two-dimension cutting problem, Dynamic programming, Defect, NP-hard.

## 1. Introduction

The two-dimensional cutting problem with defects is a research hotspot of combinatorial optimization. In the industrial manufactural area, many 2D cutting problems will encounter defects. For example, in the furniture industry, the wood panels may contain damaged areas which cannot be used for furniture panel surfaces. In the steel industry, some coils may contain defects that cannot be used as construction materials. Natural products such as leather usually have cut marks, so the defective parts can hardly be used on the surface of goods. In the literatures, the existing algorithms [1-2] for defect free problems are relatively extensive, and the research on multiple defects and guillotine cut has attracted more and more attention in recent years.

Experts have proposed many algorithms on the two-dimensional cutting problem without defect. In the latest literature, Wang et al. (2017) [2] propose a heuristic search algorithm based on grouping rules, which designs the key complement of the large and small parts division strategy and the quick recommendation of the block. Song et al. (2010) [3] propose a heuristic algorithm based on dynamic programming, which uses a subset of all possible cutting pattern and is an incomplete algorithm. Wuttke and Heese(2017) [4] propose a sequential heuristic with feedback loop and formulate the sequencing problem as a mixed integer program in the two-dimension cutting problem. They use real data to test their heuristic and illustrate its applicability to a problem of realistic size. Yoon et al. (2013) [5] propose an improved version of the cutting problem for solving standard two-dimensional cutting problem, and their algorithm removes the dominated patterns efficiently and avoids duplicated patterns. Herz (1972) [6] uses a discretization set of all necessary cutting positions to propose an accurate recursive process. Beasley (1985) [7] shows how to improve the performance of the recursive process Herz's discretization sets and introduces a heuristic correction of the algorithm which limits the number of cuts in the discretization sets.

Now there have been more and more literatures on the issue of the two-dimensional cutting problem with defects. Carnieri et al. (1993) [8] propose a heuristic dynamic programming algorithm including branch and bound search, but they only study the two-dimensional cutting problem with one defect. Vianna and Arenales (2006) [9] re-examine this problem by providing an AND/OR-based branch-qualification algorithm that further introduce a heuristic search that combines depth-first search and depth-limiting and hill-climbing strategies. Neidlein and Wäscher (2008) [10] reduce the size of discretization sets in the algorithm proposed by Vianna and Arenales (2006) [9], however, their algorithms do not obtain optimal solutions. Afsharian et al. (2014) [11] modify the predecessor's heuristic dynamic programming algorithm to solve the problem with 4 defects. Their discretization sets size are cumbersome, which means the computational efficiency is not high. Martin M. et al. (2019) [12] propose a compact integer linear programming (ILP) model for the problem based on the discretization of the defective object and develop a Benders decomposition algorithm and a constraint-programming (CP) based algorithm as solution methods. For the non-guillotine cutting problem, Gonçalves and Wäscher (2020) [13] combine a MIP model with a new hybrid algorithm to solve it and Birgin et al. (2020) [14] propose a mixed integer linear programing model for the problem with usable leftovers. Velasco and Eduardo (2019) [15] study the constrained two-dimensional guillotine cutting problem for obtaining upper bounds. Russo et al. (2020) [16] review the best exact and heuristic solutions for C2DC and reviewed and classified the available upper bound. Wu et al. (2019) [17] discuss the same problem but They don't publish the source of their data.

In this paper, inspired by the previous algorithms [7,11], an improved heuristic dynamic programming algorithm is proposed to solve the problem with multiple defects in the way of guillotine cut. The algorithm reduces the discretization sets size of Afsharian et al. (2014) [11]. However, the cut positions at one unit from the four boundaries of the defect are added to the new discrete set. The computational results show that the algorithm improves the computational efficiency on thousands of typical instances.

Section 2 of the paper presents a description of the problem. Section 3 gives a detailed description of the algorithm and prove two important theorems about the complexity of the algorithm. Section 4 gives the calculation results of thousands of typical examples, and compares the algorithm in this paper with the best algorithm at present. Section 5 draws the conclusion.

## 2. Problem Description

For the convenience of later description, table 1 shows a list of the symbols with their meanings to be used.

**Table 1.** The list of the symbols with their meanings

| symbol | meaning | symbol | meaning |
|---|---|---|---|
| $W_0$ | the width of the large object | $H_0$ | the height of the large object |
| $w_i^s$ | the width of $i$-th small rectangular block | $h_i^s$ | the height of $i$-th small rectangular block |
| $d_j$ | the $j$-th defect | $p_i$ | the number of cuts of $i$-th small rectangular block |
| $w_j^d$ | the width of $j$-th defect | $h_j^d$ | the height of $j$-th defect |
| $x_j^d$ | the $x$-axis of $j$-th defect | $y_j^d$ | the $y$-axis of $j$-th defect |
| $z_x$ | the vertical cutting position | $z_y$ | the horizontal cutting position |
| $\Phi_x^C(x)$ | the vertical discretization set of the C-block | $\Phi_y^C(y)$ | the horizontal discretization set of the C-block |
| $\Phi_x^D(x)$ | the vertical discretization set of the D-block | $\Phi_y^C(y)$ | the horizontal discretization set of the D-block |
| $v_i$ | the value of $i$-th small rectangular block | | |

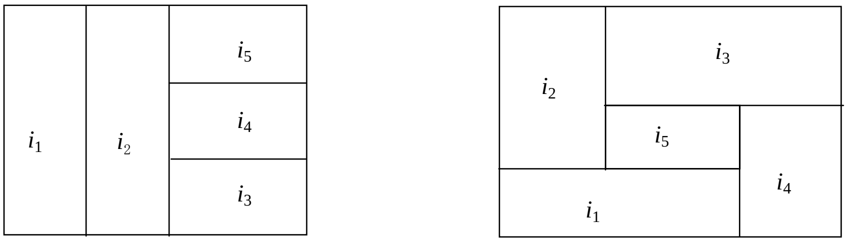
Let $m$ different types of small rectangular blocks $i(i = 1, 2, \dots, m)$, each associated with an integer width $w_i^s$, an integer height $h_i^s$ and the profit value $v_i$, must be cut from a single rectangular large object with a width of $W_0$ and a height of $H_0$ to maximize the total value of the small rectangular blocks produced by the cutting process. The solution of the problem is a cutting pattern, a form of small rectangular blocks produced from large object and a description of the layout in which the small rectangular blocks are arranged on large object. In this layout, all small rectangular blocks must be arranged parallel to the large object which is called a feasible solution of the problem. To establish the Cartesian coordinate system, let the bottom-left vertex of the large object be at the origin, the $x$-axis and $y$-axis be coincident with the wide and high edges of the object respectively. So, the large object can be represented by $(0, 0, W_0, H_0)$. For the issues to be considered, the following constraints should be met:

- The number of each type of small rectangular block cut is unlimited, that is $p_i \geq 0$ ;
- When cutting any small rectangular blocks, the given length and width orientation must be maintained, and 90 ° rotation is not allowed;
- Every cutting action must be guillotine mode, i.e., each cutting action exactly divides the current sheet into two parts (see Fig. 1);
- Every small rectangular block cut from the large object can not contain any defects, and its lower left coordinate shall be $(x_i^s, y_i^s)$, that is $0 \leq x_i^s \leq W_0 - w_i^s, 0 \leq y_i^s \leq H_0 - h_i^s$ , it must meet the requirements: $(x_j^d + w_j^d \leq x_i^s)\ or\ (x_j^d \geq x_i^s + w_i^s)\ or\ (y_j^d + h_j^d \leq y_i^s)\ or\ (y_j^d \geq y_i^s + h_i^s)$.

The cutting problem solved in this paper requires that all of the above constraints to be satisfied, that is, it is a two-dimensional, unconstrained, guillotine, single large object cutting patterns problem with defects (2D_UG_SLOPP_D). 2D_UG_SLOPP_D is generalization of 2D_UG_SLOPP.

Let $P$ be a feasible solution of the problem and $p_i \geq 0$ $(i = 1, 2, \dots, m)$ be the amount of the $i$-th small rectangular block cut from the large object in $P$, then $(p_1, p_2, \dots, p_m)$ is use to describe the feasible cutting pattern in this paper. The goal of the problem is to maximize the value of the small blocks cut from the large object, and the object function of the problem can be expressed as follows:

$$\begin{cases} \max\ V = \sum_{i=1}^{m} v_i * p_i \\ s.t.\ (p_1, p_2, \ldots, p_m)\ is\ a\ feasible\ cutting\ pattern \end{cases} \qquad (\mathbf{1})$$



(a) A guillotine cutting mode

(b) A Non-guillotine cutting mode

**Fig. 1.** Two cutting modes: guillotine and non-guillotine

A defect is actually an irregular figure. Considering the cutting method here, it is appropriate to use rectangular area to express defects. Let *n* defects be in the large object, the *j*th defect $d_j (j = 1, 2, \ldots, n)$ have a width $w^{d}_{,j}$ and a height $h^{d}_{,j}$, and its bottom-left vertex on the large plate be $(x_j^d, y_j^d)$, then $d_j$ can be represented by $(x_j^d, y_j^d, w_j^d, h_j^d)$ (see Fig. 2).

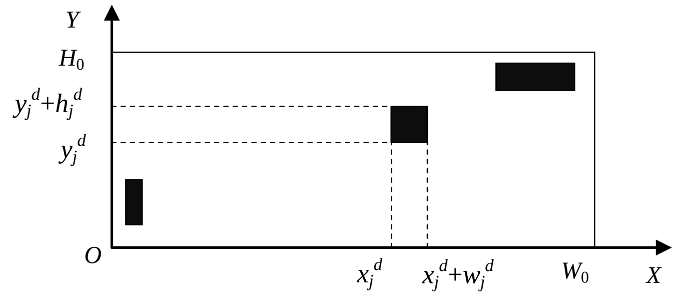


**Fig. 2.** The defects on the large object and their representation

## 3. Algorithm Description

The algorithm in this paper is called an improved heuristic dynamic programming algorithm (IHDP). It combines quasi human idea with dynamic programming algorithm. Using dynamic programming algorithm to solve 2D_UG_SLOPP_D, the resulting subproblem is either 2D_UG_SLOPP_D or 2D_UG_SLOPP. Adopting different methods to solve these two different problems is the critical improvement of this algorithm.

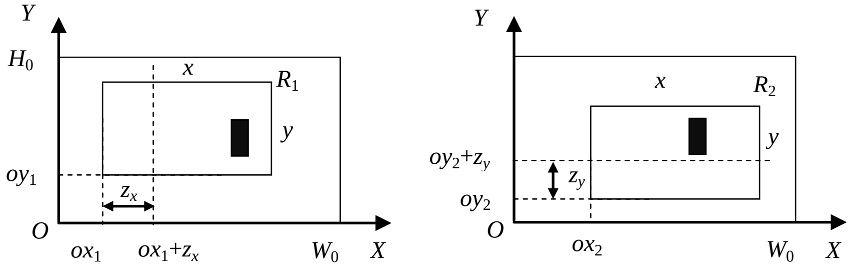
### 3.1. Basic definition

For the convenience of the following description, here are two important definitions.

**Definition 1** (**sub-block**). In the guillotine mode, multiple rectangles which are neither the small rectangular blocks nor the wastes are formed after the large plate is cut several times. These rectangles are called sub-block. In this paper, the large object is regarded as the largest sub-block, and a sub-block corresponds to a sub-problem and vice versa.

Let the coordinates of a sub-block be ($ox$, $oy$), the width and the height of it be $x$ and $y$ respectively, then the sub-block is represented by $R$=($ox$, $oy$, $x$, $y$).

According to this definition, two sub-blocks $R_1$=($ox_1$, $oy_1$, $x$, $y$) and $R_2$=($ox_2$, $oy_2$, $x$, $y$) with different coordinates in the sub-block are different sub-problems, even if they have the same size. For a vertical cut (parallel to the $y$-axis) on the sub-block $R_1$=($ox_1$, $oy_1$, $x$, $y$) at the cut position $z_x$, two smaller sub-blocks are ($ox_1$, $oy_1$, $z_x$, $y$) and ($ox_1$+$z_x$, $oy_1$, $x$-$z_x$, $y$). Similarly, for a horizontal cut (parallel to the $x$-axis) at the cut position $z_y$ on $R_2$, two sub-blocks ($ox_2$, $oy_2$, $x$, $z_y$) and ($ox_2$, $oy_2$+$z_y$, $x$, $y$-$z_y$) are also formed (see Fig. 3).



(a) a vertical cut on a sub-block

(b) a horizontal cut on a sub-block

**Fig. 3.** The guillotine cut on the sub-block (vertical or horizontal) results in two smaller sub-blocks

**Definition 2** (**C-block and D-block**). For a sub-block $R$=($ox$, $oy$, $x$, $y$), if it contains a defect or overlap with any defect, it is called a D-block; otherwise it is called a C-block.

### 3.2. Discretization Sets

The cutting position set on the sub-block is a discretization set. If the sub-block is a C-block, the discretization sets [8] are defined by the equations (2), (3), and (4). Otherwise, the discretization sets are defined by the equations (5), (6), and (7). Z+ belongs to a positive integer set.

**Discretization sets of C-blocks.** If a sub-block is a C-block, the discretization sets defined by the equations (2), (3), and (4) are quite same with the discretization sets proposed by Afsharian et al. [11]. They are established by the solution of a one-dimensional knapsack problem with the width and height of the small rectangular blocks. Let $\boldsymbol{\Phi_x^C(x)}$ and $\boldsymbol{\Phi_y^C(y)}$ denote the vertical discretization set and the horizontal discretization set of sub-blocks respectively, they are described as follows:

$$\Phi_x^C(x) = \left\{ z_x | z_x = \sum_{i=1}^{m} \alpha_i w_i^s , 1 \le z_x \le x - w_0/2, \alpha_i \in Z^+ \cup \{0\}, \forall i \right\} \qquad (\mathbf{2})$$

$$\Phi_y^C(y) = \left\{ z_y | z_y = \sum_{i=1}^{m} \beta_i h_i^s , 1 \le z_y \le y - h_0/2, \beta_i \in Z^+ \cup \{0\}, \forall i \right\} \qquad (\mathbf{3})$$

$$w_0 = min\{w_i^s : h_i^s < y\} \ \ , h_0 = min\{h_i^s : w_i^s < x\} , i = 1, 2, \dots, m \qquad (\mathbf{4})$$

**Discretization sets of D-blocks.** If a sub-block is a D-block, the discretization sets defined by the equations (5), (6), and (7). These discretization sets add the cutting position of one unit outside four boundaries of each defect into $\boldsymbol{\Phi_x^C(x)}$ and $\boldsymbol{\Phi_y^C(y)}$. They reduce the discretization sets proposed by Afsharian et al. [11].

$$\Phi_x^D(x) = \Phi_x^C(x) \cup \left\{ \begin{matrix} z_x | z_x = \delta_j\left(x_j^d - ox - 1\right) \, or \, \gamma_j\left(x_j^d + w_j^d - ox + 1\right), \\ \delta_j, \gamma_j \in \{0,1\}, \ j = 1, 2, \dots, n \end{matrix} \right\} \qquad (\mathbf{5})$$

$$\Phi_y^D(y) = \Phi_y^C(y) \cup \left\{ \begin{matrix} z_y | z_y = \mu_j\left(y_j^d - oy - 1\right) \, or \, \theta_j\left(y_j^d + h_j^d - oy + 1\right), \\ \mu_j, \theta_j \in \{0,1\}, \ j = 1, 2, \dots, n \end{matrix} \right\} \qquad (\mathbf{6})$$

$$w_0 = min\{w_i^s : h_i^s < y\} \ \ , h_0 = min\{h_i^s : w_i^s < x\} , i = 1, 2, \dots, m \qquad (\mathbf{7})$$

### 3.3. Dynamic programming

This algorithm is an improved heuristic algorithm based on dynamic programming. For subproblems without defects (C-block), IHDP uses the method of Herz (1972) [6] and Beasley (1985) [7] to construct recursive function *F* (*x*, *y*) for solving it. In this paper, the upper bound of discretization set is extended to *x* - $w_0$/2， *y* - $h_0$/2. For the sub-problem with defects (D-block), IHDP adopts a denormalization recursive function *F*(*ox*, *oy*, *x*, *y*) which is different against Afsharian et al. [11] to deal with it. Furthermore, the cutting positions of one unit outside four boundaries of each defect are added into the discretization sets.

It is easy to get a lower bound of the objective function of C-blocks which can be got by the function $g(x, y)$. Every time, the sub-blocks are divided into the same type small rectangular blocks, and the lower bound is the maximum value of *m* cutting pattern. The functions *F* (*x*, *y*) and $g(x, y)$ are as follows:

$$g(x, y) = max\left\{ v_i \cdot \left\lfloor \frac{x}{w_i^s} \right\rfloor \cdot \left\lfloor \frac{y}{h_i^s} \right\rfloor \right\}, w_i^s < x, h_i^s < y, i = 1,2, \dots, m \qquad (\mathbf{8})$$

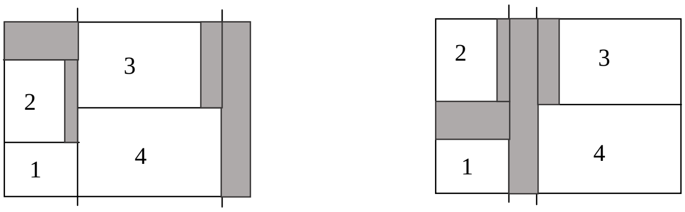
$$F(x,y) = max\begin{Bmatrix} g(x,y), \\ F(z_x,y) + F(p(x-z_x),y),\ z_x \in \Phi_x^C(x), 1 \le z_x \le \frac{x}{2}, \\ F(x,z_y) + F\left(x, q(y-z_y)\right), z_y \in \Phi_y^C(y), 1 \le z_y \le \frac{y}{2}, \\ x \in \Phi_x^C(W_0) \cup \{W_0\},\ \ y \in \Phi_y^C(H_0) \cup \{H_0\} \end{Bmatrix} \qquad (\mathbf{9})$$

Where, if $x < w_0\ or\ y < h_0$, then $F(x,\ y)$=0. Due to the appearance of the repeated cutting pattern, the discretization set of vertical (or horizontal) in (9) is limited to half the width (height) of the sub-block. In addition, Beasley (1985) [7] has proved that a kind of normalized cutting pattern will not result in the optimization of the solution of the recursive equation to solve the C-block. This pattern is to arrange the waste at the bottom left of the C-block (see Fig. 4), which is implemented with the two functions $p(x)$ and $q(y)$ introduced in the above recursive function. These two functions are described as follows:

$$p(x) = max\{0, z_x\}, z_x \le x, z_x \in \Phi_x^C(W_0), x < W_0, p(W_0) = W_0 \qquad (\mathbf{10})$$

$$q(y) = max\{0, z_y\}, z_y \le y, z_y \in \Phi_y^C(H_0), y < H_0, q(H_0) = H_0 \qquad (\mathbf{11})$$

$p(x)$ represents the cut position nearest to the sub-block width $x$, and correspondingly, $q(y)$ is the cut position nearest to the sub-block height $y$.



(a) Normalized cutting pattern

(b) Non-normalized cutting pattern

**Fig. 4.** Normalized and Non-normalized cutting pattern (shaded parts are scrap)

Based on the above recursive function, we design the following Solver to solve the subproblem C-block.

Here is the description of the Solver：

```
Solver: Algorithm for solving 2D_UG_SLOPP
  Input:  Subproblem $R=(x, y)$, $\Phi_x^C(x)$, $\Phi_y^C(y)$
  Output: Cutting Pattern recorded as $F(x, y)$
If ($R=(x, y)$ has been solved)  Then
  Return $F(x, y)$;
  Else If ($x<w_0$ or $y<h_0$)
    Return 0;
    Else
      $F^*(x, y)=0$;
      For($z_x \in \Phi_x^C(x)$, $1 \le z_x \le x/2$)
        $F^*(x, y) \leftarrow$ max(Solver($R=(z_x, y)$) + Solver ($R=(p(x-z_x), y)$), $F^*(x, y)$);
      End for
```

```
        $F^{**}(x, y)=0$;
      For($z_y \in \Phi_y^C(y)$, $1 \le z_y \le y/2$)
        $F^{**}(x, y) \leftarrow$ max(Solver($R=(x, z_y)$) + Solver ($R=(x, q(y-z_y))$), $F^{**}(x, y)$);
        End for
      Return $F(x, y)$=max($g(x, y)$, $F^{*}(x, y)$, $F^{**}(x, y)$)
    End if
  End if
```

Obviously, because the original problem 2D_UG_SLOPP_D has defects, and the Solver is only called in the process of solving the original problem, which means that neither the $x$ nor $y$ in the subproblem $R=(x, y)$ is known in the original problem, the input here is not the initial value of the problem.

The equation (12) is used to determine the recursive function of the optimal cutting pattern with the D-block $R=(ox, oy, x, y)$:

$$F(ox,oy,x,y)=\begin{cases} F(x,y), \quad if\ R=(ox,oy,x,y)\ is\ C-block; \\ max\left\{\begin{array}{c} F(ox,oy,z_x,y)+F(ox+z_x,oy,x-z_x,y), \\ F(ox,oy,x-z_x,y)+F(ox+x-z_x,oy,z_x,y), \\ F(ox,oy,x,z_y)+F(ox,oy+z_y,x,y-z_y), \\ F(ox,oy,x,y-z_y)+F(ox,oy+y-z_y,x,z_y), \\ z_x \in \Phi_x^D(x), z_y \in \Phi_y^D(y)\ , \\ w_0 \le x \le W_0, h_0 \le y \le H_0, \\ 0 \le ox \le W_0, 0 \le oy \le H_0, \end{array}\right\}, \\ \qquad if\ R=(ox,oy,x,y)\ is\ D-block. \end{cases} \qquad \mathbf{(12)}$$

Where, if $x < w_0\ or\ y < h_0$, then $F$(ox, oy, x, y).

The algorithm in this paper adopts denormalization strategy to solve the problem 2D_UG_SLOPP_D. In fact, the locations of the defects on D-block are uncertain, which means that this kind of normalization treatment may waste some plates, thus reducing the chance of obtaining the optimal solution. For example, if the defect is located in the lower left corner of a D-block, the more the upper right area of the block is used, the better the solution is possible. Based on all these favorable practices, we develop the advanced algorithm of predecessors [9, 10, 18] and get the improved heuristic dynamic programming algorithm IHDP.

Here is the description of the IHDP：

```
IHDP：Algorithm for solving 2D_UG_SLOPP_D

  Input：Subpoblem $R=(ox, oy, x, y)$, $\Phi_x^D(x)$, $\Phi_y^D(y)$, Defects set $\{d_1, d_2, \ldots, d_n\}$
  Output：Cutting pattern，Recorded as $F(ox, oy, x, y)$
If ($R=(ox, oy, x, y)$ has been solved)  Then
  Return $F(ox, oy, x, y)$;
  Else If ($x<w_0$ or $y<h_0$)
    Return $F(ox, oy, x, y)=0$;
    Else If ($R=(ox, oy, x, y)$ is $C$-block)
      Return $F(x, y)$=Solver($R=(x, y)$);
      Else   // $R=(ox, oy, x, y)$ is $D$-block
        $F^{*}(ox, oy, x, y)=0$;
        For($z_x \in \Phi_x^D(x)$, $1 \le z_x \le x-1$)
          $F^{(1)}(ox, oy, x, y)$=IHDP($R=(ox, oy, z_x, y)$)+IHDP($R=(ox+z_x, oy, x-z_x, y)$);
          $F^{(2)}(ox, oy, x, y)$=IHDP($R=(ox, oy, x-z_x, y)$)+IHDP($R=(ox+x-z_x, oy, z_x, y)$);
          $F^{*}(ox, oy, x, y) \leftarrow$ max($F^{(1)}(ox, oy, x, y)$, $F^{(2)}(ox, oy, x, y)$, $F^{*}(ox, oy, x, y)$);
```

```
13        End for
14        $F^{**}(ox, oy, x, y)=0$;
15        For($z_y \in \Phi_y^D(y)$, $1 \le z_y \le y-1$)
16          $F^{(3)}(ox, oy, x, y)$=IHDP($R=(ox, oy, x, z_y)$)+IHDP($R= (ox, oy+z_y, x, y-z_y)$);
17          $F^{(4)}(ox, oy, x, y)$=IHDP($R=(ox, oy, x, y-z_y)$) +IHDP($R= (ox, oy+y-z_y, x, z_y)$);
18          $F^{**}(ox, oy, x, y) \leftarrow$ max($F^{(3)}(ox, oy, x, y)$, $F^{(4)}(ox, oy, x, y)$, $F^{**}(ox, oy, x, y)$);
19         End for
20         Return $F(ox, oy, x, y)$= max($F^{*}(ox, oy, x, y)$, $F^{**}(ox, oy, x, y)$);
21      End if
22    End if
23  End if
```

IHDP is implemented to solve the original problem 2D_UG_SLOPP_D and the subproblems with defects. Only when the subproblem is of type 2D_UG_ SLOPP, the Solver is called in line 6 to solve it. So, the initial values of the algorithm are $R_0 = (0, 0, W_0, H_0)$, $\Phi_x^D(W_0)$, $\Phi_y^D(H_0)$ and all the defects $d_1, d_2, \ldots, d_n$.

### 3.4. Algorithm complexity

In this section, we study the computational aspects of the algorithm. We analyze the time complexity in the worst case and get an estimation of pseudo polynomials.

**Theorem 1** The time complexity in the worst case of the improved heuristic dynamic programming for solving the 2D_UG_SLOPP_D is:

$$O\left(|\Phi_x^D(W_0)| \cdot |\Phi_y^D(H_0)| \cdot \left(|\Phi_x^D(W_0)| + |\Phi_y^D(H_0)|\right)\right) \quad (\mathbf{13})$$

**Proof** For a given single large object $(W_0, H_0)$, the recursive function requires $O(|\Phi_x^D(W_0)| \cdot |\Phi_y^D(H_0)|)$ operations for each iteration. Therefore, the calculation involves a total of time complexity as equation (13).

**Theorem 2** Let $w_0 = \min\{w_i^s\}$ and $h_0 = \min\{h_i^s\}, i = 1,2,\ldots,m$. And let $\rho_w = \lfloor W_0/w_0 \rfloor \le W_0/w_0, \rho_h = \lfloor H_0/h_0 \rfloor \le H_0/h_0$, then:

$$|\Phi_x^D(W_0)| = \sum_{t=1}^{\rho_w} C_{t+m-1}^t + 2n \quad (\mathbf{14})$$

$$|\Phi_y^D(H_0)| = \sum_{t=1}^{\rho_h} C_{t+m-1}^t + 2n \quad (\mathbf{15})$$

$$C_{t+m-1}^t = \frac{(t+m-1)!}{t!\,(m-1)!} \quad (\mathbf{16})$$

**Proof** By definition, each element in $\Phi_x^D(W_0)$ is a viable combination of the length of a small rectangular block $\sum_{t=1}^{\rho_w}(\sum_{i=1}^m w_i^s)^t \le W_0$ calculates the same structure of the number of terms in the polynomial. In order to obtain $\sum_{i=1}^m \alpha_i w_i^s$, in the above polynomial, $t$ must take 1 to$\rho_w$. That's because if $t > \rho_w$, then there is$\sum_{i=1}^m \alpha_i w_i^s > W_0$.

And $\sum_{t=1}^{\rho_w}(\sum_{i=1}^{m} w_i^s)^t = \sum_{t=1}^{\rho_w} C_{t+m-1}^t$. Add the above function (14) to the left and right edges of defects; similarly, the same is true for the function (15).

## 4. Calculation results and analysis

The algorithm used in this paper is implemented in the C/C++ programming language. The configuration of the computer used is: processor--Intel(R), Core (TM) i7cpu @360HZ, RAM 8GB, 64bit operator.

Three typical classes examples are included in this experiment. The first class is 14 instances in which 8 instances proposed by Carnieri et al. (1993) [8] include a single defect and 6 instances proposed by Vianna and Arenales (2006) [9] include multiple defects. The original object's width $W_0$ = 200, height $H_0$ = 100, and 5 types of small rectangular blocks. The other two classes are generated by Neidlein's [18] instance generator. One class of them has already been generated and adopted both by both Afsharian et al. [11] and by Martin et al. [12]. Another class is generated by ourselves, and the seed values of random numbers are 3, 7 and 11, respectively.

In this paper, five other typical algorithms are selected to compare their effectiveness and efficiency with IHDP. The **o**bjective **f**unction **v**alues (OFV) obtained by all these algorithms is the important index. Furthermore, according to the literature [18], the algorithm DPC (dynamic programming with complete discretization set) is implemented. Another index, GAP = ($OFV_{DPC}$-$OFV_{IHDP}$)/ ($OFV_{DPC}$ + $10^{-10}$)*100, is used to present these algorithms' performance.

### 4.1. International Samples

In Table 2, The algorithm is compared with other three algorithms. Neidlein et al. (2008) [10] only computes the instances with a single defect and obtains the optimal solutions of 5 out of 8 instances. Vianna and Arenales [9] gets an optimal solution of the instance with multiple defects, however, it does not public their computation time. Both Afsharian et al. [11] and IHDP obtain the 14 optimal solutions in a short time.

**Table 2.** Operation results and comparison of four algorithms

| Ins. | IHDP | | Afsharian etc.(2014) | | Vianna etc.(2006) | | Neidlein etc.(2008) | |
|---|---|---|---|---|---|---|---|---|
| | OFV | Comp. Time(s) | OFV | Comp. Time(s) | OFV | Comp. Time(s) | OFV | Comp. Time(s) |
| A1 | **166** | 0.86 | **166** | 18.86 | **166** | 4.61 | **166** | 0.52 |
| A2 | **166** | 0.67 | **166** | 16.43 | 160 | 3.57 | 160 | 0.77 |
| A3 | **166** | 0.74 | **166** | 16.47 | 162 | 4.40 | 162 | 1.77 |
| A4 | **164** | 0.25 | **164** | 18.25 | 160 | 3.15 | 160 | 0.27 |
| A5 | **164** | 0.31 | **164** | 76.96 | **164** | 13.51 | **164** | 4.11 |
| A6 | **164** | 0.47 | **164** | 0.90 | **164** | 1.32 | **164** | 1.44 |
| A7 | **158** | 0.21 | **158** | 0.81 | **158** | 12.47 | **158** | 1.07 |
| A8 | **154** | 0.13 | **154** | 1.21 | **154** | 8.07 | **154** | 0.50 |
| A9 | **160** | 0.87 | **160** | 14.32 | 153 | - | - | - |
| A10 | **158** | 0.49 | **158** | 2.22 | 148 | - | - | - |
| A11 | **151** | 6.29 | **151** | 26.78 | 143 | - | - | - |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| A12 | **156** | 119.30 | **156** | 1126.44 | 150 | - | - | - |
| A13 | **150** | 2.19 | **150** | 9.06 | 142 | - | - | - |
| A14 | **160** | 0.09 | **160** | 1.00 | **160** | - | - | - |

Note: Bold type in the table represents the optimal solution

In addition, the optimal layout the two instances A11 and A12 which have 4 and 5 defects respectively are given below (see Fig. 5-6).

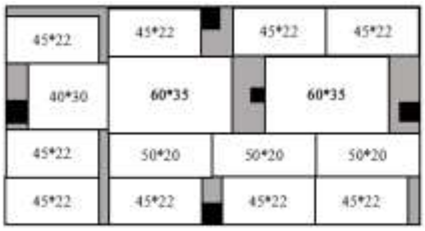


**Fig. 5.** A11 cutting pattern result

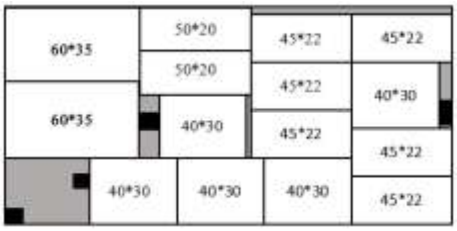


**Fig. 6.** A12 cutting pattern result

## 4.2. Randomly generated instances

These examples are set by the instance generator procedure of Neidlein and Wäscher (2016) [18]. With respect to the size of the large plate, it is distinguished between two categories of problem instances, i.e., quadratic and non-quadratic. In small category, the size of the large object is fixed to 5,625 square units, and the size of the large object is fixed to 22,500 square units in medium category. Furthermore, the two different shapes of the large plate considered for each category are quadratic (75, 75) and non-quadratic (112, 50) for the small category and quadratic (150, 150) and non-quadratic (225, 100) for medium category, respectively. The number of types of small rectangles is set to 5. 10, 15, 20 and 25. The type width and height of the small rectangular block are uniformly obtained from $[W_0/\varpi, 3W_0/4]$ and $[H_0/\varpi, 3H_0/4]$, respectively, where $\varpi$ in all categories is 6, 8, and 10. The defect is set to 1~4, and 15 examples of each defect are averaged for a group. The width and height of the defect are uniformly obtained from the ranges $[W_0/10, W_0/6]$ and $[H_0/10, H_0/6]$, respectively. The position of each defect is represented by the position of the defect in the lower left corner of the large rectangular block, generated using a uniform distribution in the range of $[0, W_0 - w_0]$ and $[0, H_0 - h_0]$, and then these values are rounded. So, the instances number of each size category are 2×5×3×15×4=1800.

**Instances in the literatures.** The instances in Table 3 and Table 4 are generated by Afs -harian et al. (2014) [11] without providing the value of their random seed, however, the data can be obtained from the following website: www. dep.ufscar.br/docentes/munari/ cuttingpacking/.

In Table 3, the other three algorithms are the best ones we have got. As a complete DP (Dynamic programming takes every integer both in [1, $W_0$] and in [1, $H_0$] as cutting point) [18], DPC does obtain the optimal solutions of all instances in guillotine manner. IHDP obtains the optimal solution of all cases, however, DPD and B&BC get the optimal solutions of one group and ten groups of instances, respectively. However, The

Gap of B&BC on the average of 30 groups is more than twice that of DPD, so the stability of the latter is better than the former. B&BC takes far more time than the other three algorithms.

In Table 4, Martin et al. [12] doesn't show the calculation details of the 30 group instances, so the computation results of DPC, DPD and IHDP are compared. None of the optimal solutions of these instances are obtained by DPD. However, IHDP gets 26 out of 30 groups optimal solutions of the instances, and the four GAPs are very close to 0.00. Here, although the computation time difference among them is not very big, IHDP takes more time than DPD, but less than DPC.

**New randomly generated instances.** In order to further test the performance of IHDP, the generator of Neidlein et al. (2016) [18] is used to generate new data. Table 5 and Table 6 are instances of small and medium scale, and the value of their random number seed =3. Table 7 and Table 8 are small scale instances, and the value of their random number seed=7, 11, respectively (randomly selected from the uniform distribution of [1, 12]). Because these are new data, only DPC and IHDP are used for comparison.

Table 5 shows the computation results of small-scale instances. Both IHDP and DPC get the optimal solution of all cases, but IHDP takes almost a tenth of DPC. Table 6 shows the computation results of the medium scale case. IHDP does not get the optimal solution on five groups of instances. However, the result of IHDP is very close to the optimal solution, and its final average Gap is still 0.000. Moreover, it takes less than half the computing time of DPC.

Both in Table 7 and Table 8, 1800 small scale instances are generated by seed values of 7 and 11, respectively. In Table 7, IHDP obtains the optimal solutions of 28 out of 30 groups of the instances, but the GAP is only 0.2%. In Table 8, IHDP obtains all the optimal solution of the instances. In both cases, IHDP takes about half as long as DPC.

According to the results shown in Tables 3 to 8 above, IHDP performs stablely on the instances generated by both ourselves and Afsharian et al. [11], and it is an effective and efficient algorithm to solve the original problem 2D_UG_SLOPP_D in guillotine mode.

## 5. Conclusion

In this paper, the smaller discretization sets are constructed to solve the two-dimensional cutting problem with defects in guillotine manner. Especially, an improved heuristic-dynamic programming algorithm is proposed, which adapts two different methods to for the subproblems 2D_UG_SLOPP_D and 2D_UG_SLOPP, respectively. Almost all the optimal solutions of over ten thousand typical instances are obtained. An important theorem on its complexity of the algorithm is proved. Future research could focus on solving the larger scale instance or on modifying the discretization set definition or on solving different type of cutting problem such as involving the constraints of the largest number of each type of the small rectangular block.

**Table 3.** Four algorithms comparison on the small size 1800 instances of Afsharian et al.[11]

| instances | $m$ | $\varpi$ | DPC | | DPD | | | B&BC | | | IHDP | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) | OFV | time(s) | GAP(%) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | **3694.31** | 11.45 | **3694.31** | 3.52 | 0.00 | **3694.31** | 0.05 | 0.00 | **3694.31** | 0.41 | 0.00 |
| 2 | 10 | 6 | **4256.01** | 12.12 | 4229.85 | 4.52 | 0.61 | **4256.01** | 15.20 | 0.00 | **4256.01** | 2.50 | 0.00 |
| 3 | 15 | 6 | **4566.10** | 13.33 | 4540.38 | 5.36 | 0.56 | **4566.10** | 1.57 | 0.00 | **4566.10** | 3.92 | 0.00 |
| 4 | 20 | 6 | **4615.43** | 13.09 | 4593.41 | 5.76 | 0.48 | 4605.70 | 42.80 | 0.21 | **4615.43** | 5.32 | 0.00 |
| 5 | 25 | 6 | **4694.01** | 13.89 | 4674.66 | 6.19 | 0.41 | 4691.43 | 56.70 | 0.05 | **4694.01** | 6.31 | 0.00 |
| 6 | 5 | 8 | **3683.23** | 11.14 | 3638.03 | 2.86 | 1.23 | **3683.23** | 0.87 | 0.00 | **3683.23** | 0.60 | 0.00 |
| 7 | 10 | 8 | **4386.16** | 12.94 | 4375.50 | 4.74 | 0.24 | 4372.03 | 62.00 | 0.32 | **4386.16** | 3.20 | 0.00 |
| 8 | 15 | 8 | **4613.23** | 14.07 | 4585.26 | 6.09 | 0.61 | 4604.15 | 78.03 | 0.19 | **4613.23** | 5.26 | 0.00 |
| 9 | 20 | 8 | **4883.78** | 14.69 | 4876.38 | 6.27 | 0.15 | 4850.66 | 150.20 | 0.67 | **4883.78** | 6.83 | 0.00 |
| 10 | 25 | 8 | **4835.60** | 14.48 | 4826.93 | 7.76 | 0.18 | 4792.66 | 91.92 | 0.88 | **4835.60** | 7.49 | 0.00 |
| 11 | 5 | 10 | **4083.56** | 12.71 | 4055.10 | 3.18 | 0.70 | 4001.58 | 16.56 | 2.00 | **4083.56** | 1.11 | 0.00 |
| 12 | 10 | 10 | **4710.91** | 14.60 | 4686.56 | 5.69 | 0.52 | 4637.40 | 107.65 | 1.56 | **4710.91** | 5.39 | 0.00 |
| 13 | 15 | 10 | **4845.16** | 14.78 | 4826.68 | 6.74 | 0.38 | 4678.93 | 122.56 | 3.43 | **4845.16** | 6.25 | 0.00 |
| 14 | 20 | 10 | **4928.65** | 15.25 | 4920.95 | 9.35 | 0.16 | 4629.06 | 155.83 | 6.07 | **4928.65** | 8.20 | 0.00 |
| 15 | 25 | 10 | **4968.50** | 16.13 | 4960.23 | 10.71 | 0.17 | 4783.45 | 217.93 | 3.72 | **4968.50** | 9.44 | 0.00 |
| 16 | 5 | 6 | **3530.41** | 10.74 | 3480.63 | 2.61 | 1.41 | **3530.41** | 0.03 | 0.00 | **3530.41** | 0.26 | 0.00 |
| 17 | 10 | 6 | **4339.18** | 11.53 | 4305.43 | 4.31 | 0.78 | **4339.18** | 20.38 | 0.00 | **4339.18** | 2.45 | 0.00 |
| 18 | 15 | 6 | **4495.20** | 14.54 | 4470.76 | 4.31 | 0.54 | **4495.20** | 13.40 | 0.00 | **4495.20** | 4.05 | 0.00 |
| 19 | 20 | 6 | **4618.16** | 14.66 | 4607.75 | 5.80 | 0.23 | 4611.68 | 59.53 | 0.14 | **4618.16** | 5.77 | 0.00 |
| 20 | 25 | 6 | **4686.21** | 15.42 | 4679.73 | 6.78 | 0.14 | 4671.68 | 82.28 | 0.30 | **4686.21** | 6.75 | 0.00 |
| 21 | 5 | 8 | **3829.66** | 13.69 | 3752.31 | 2.03 | 2.02 | **3829.66** | 7.83 | 0.00 | **3829.66** | 0.68 | 0.00 |
| 22 | 10 | 8 | **4536.90** | 15.30 | 4523.00 | 4.56 | 0.31 | **4536.90** | 26.33 | 0.00 | **4536.90** | 4.02 | 0.00 |
| 23 | 15 | 8 | **4792.70** | 14.54 | 4766.90 | 5.65 | 0.54 | 4770.53 | 130.25 | 0.46 | **4792.70** | 6.10 | 0.00 |
| 24 | 20 | 8 | **4706.16** | 15.16 | 4685.91 | 5.92 | 0.43 | 4681.53 | 74.36 | 0.52 | **4706.16** | 6.78 | 0.00 |
| 25 | 25 | 8 | **4791.00** | 15.66 | 4772.01 | 8.09 | 0.40 | 4715.31 | 121.96 | 1.57 | **4791.00** | 7.92 | 0.00 |
| 26 | 5 | 10 | **4138.25** | 13.43 | 4076.48 | 3.63 | 1.49 | **4138.25** | 11.50 | 0.00 | **4138.25** | 0.99 | 0.00 |
| 27 | 10 | 10 | **4396.10** | 15.29 | 4357.55 | 4.53 | 0.88 | 4374.31 | 70.63 | 0.49 | **4396.10** | 3.94 | 0.00 |
| 28 | 15 | 10 | **4688.12** | 15.95 | 4661.92 | 5.70 | 0.56 | 4590.73 | 149.97 | 2.07 | **4688.12** | 6.31 | 0.00 |
| 29 | 20 | 10 | **4895.85** | 16.14 | 4891.43 | 8.71 | 0.09 | 4705.36 | 154.61 | 3.89 | **4895.85** | 8.40 | 0.00 |
| 30 | 25 | 10 | **4988.35** | 16.20 | 4975.15 | 9.31 | 0.26 | 4856.01 | 206.26 | 2.65 | **4988.35** | 8.92 | 0.00 |
| Average | | | 4506.56 | 14.10 | 4483.04 | 5.69 | 0.52 | 4456.45 | 74.97 | 1.11 | 4506.56 | 4.85 | 0.00 |

Note: Bold type in the table represents the optimal solution

**Table 4.** Comparewith DPC on the medium size 1800 instances of Afsharian et al.[11]

| instances | $m$ | $\varpi$ | DPC | | DPD | | | IHDP | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | 14794.82 | 209.58 | **14546.66** | 24.723 | 1.677 | 14794.82 | 7.89 | 0.000 |
| 2 | 10 | 6 | 15772.18 | 202.97 | **15544.76** | 35.661 | 1.442 | 15772.18 | 42.14 | 0.000 |
| 3 | 15 | 6 | 17356.57 | 146.04 | **17263.78** | 35.827 | 0.535 | 17356.57 | 77.21 | 0.000 |
| 4 | 20 | 6 | 18256.58 | 160.21 | **18128.60** | 58.636 | 0.701 | 18256.58 | 138.48 | 0.000 |
| 5 | 25 | 6 | 18824.05 | 172.32 | **18716.65** | 69.199 | 0.571 | **18701.42** | 163.10 | **0.007** |
| 6 | 5 | 8 | 14384.88 | 183.29 | **14276.08** | 27.278 | 0.756 | 14384.88 | 4.94 | 0.000 |
| 7 | 10 | 8 | 17780.28 | 280.81 | **17632.83** | 40.890 | 0.829 | 17780.28 | 74.94 | 0.000 |
| 8 | 15 | 8 | 18747.35 | 182.60 | **18615.33** | 72.122 | 0.704 | 18747.35 | 136.41 | 0.000 |
| 9 | 20 | 8 | 19315.80 | 193.07 | **19193.25** | 73.607 | 0.634 | 19315.80 | 185.78 | 0.000 |
| 10 | 25 | 8 | 19530.53 | 189.00 | **19435.88** | 131.198 | 0.485 | 19530.53 | 184.81 | 0.000 |
| 11 | 5 | 10 | 15174.78 | 237.29 | **15067.86** | 24.038 | 0.705 | 15174.78 | 9.19 | 0.000 |
| 12 | 10 | 10 | 18839.87 | 184.59 | **18775.98** | 63.492 | 0.339 | 18839.87 | 116.96 | 0.000 |
| 13 | 15 | 10 | 18747.58 | 192.31 | **18685.03** | 71.939 | 0.334 | **18611.28** | 153.12 | **0.007** |
| 14 | 20 | 10 | 19438.30 | 193.51 | **19372.35** | 99.747 | 0.339 | 19438.30 | 187.32 | 0.000 |
| 15 | 25 | 10 | 19822.68 | 217.56 | **19724.56** | 137.128 | 0.495 | 19822.68 | 216.96 | 0.000 |
| 16 | 5 | 6 | 13611.92 | 100.51 | **13420.90** | 21.29 | 1.403 | 13611.92 | 6.08 | 0.000 |
| 17 | 10 | 6 | 16938.10 | 152.79 | **16709.06** | 43.03 | 1.352 | 16938.10 | 84.64 | 0.000 |
| 18 | 15 | 6 | 17203.60 | 151.04 | **17058.16** | 45.42 | 0.845 | 17203.60 | 96.32 | 0.000 |
| 19 | 20 | 6 | 18606.00 | 181.05 | **18475.80** | 62.96 | 0.700 | 18606.00 | 141.73 | 0.000 |
| 20 | 25 | 6 | 18708.25 | 194.25 | **18591.30** | 74.73 | 0.625 | 18708.25 | 190.35 | 0.000 |
| 21 | 5 | 8 | 15217.80 | 128.29 | **15024.26** | 23.33 | 1.272 | 15217.80 | 15.61 | 0.000 |
| 22 | 10 | 8 | 17720.82 | 177.21 | **17541.91** | 53.70 | 1.010 | 17720.82 | 163.48 | 0.000 |
| 23 | 15 | 8 | 18620.30 | 196.23 | **18491.06** | 63.04 | 0.694 | **18518.05** | 143.48 | **0.005** |
| 24 | 20 | 8 | 18999.55 | 212.83 | **18814.46** | 84.30 | 0.974 | 18999.55 | 210.44 | 0.000 |
| 25 | 25 | 8 | 19382.40 | 225.39 | **19215.23** | 109.88 | 0.862 | 19382.40 | 217.22 | 0.000 |
| 26 | 5 | 10 | 15542.52 | 138.11 | **15248.20** | 35.21 | 1.894 | 15542.52 | 31.84 | 0.000 |
| 27 | 10 | 10 | 17366.72 | 179.47 | **17191.66** | 47.83 | 1.008 | 17366.72 | 106.46 | 0.000 |
| 28 | 15 | 10 | 18913.72 | 203.36 | **18733.61** | 92.64 | 0.952 | 18913.72 | 192.05 | 0.000 |
| 29 | 20 | 10 | 19632.15 | 220.12 | **19537.85** | 131.77 | 0.480 | **19513.54** | 220.01 | **0.006** |
| 30 | 25 | 10 | 19707.63 | 233.18 | **19561.21** | 122.08 | 0.743 | 19707.63 | 230.33 | 0.000 |
| Average | | | 17765.26 | 187.97 | 17619.81 | 65.89 | 0.845 | 17749.26 | 124.98 | 0.001 |

Note: Bold type in the table represents the non-optimal solution

**Table 5.** Compare with DPC on the small size 1800 new instances (seed=3)

| instances | *m* | $\varpi$ | DPC | | IHDP | | |
|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | 3825.80 | 9.82 | 3825.80 | 0.15 | 0.00 |
| 2 | 10 | 6 | 4381.25 | 12.16 | 4381.25 | 0.62 | 0.00 |
| 3 | 15 | 6 | 4567.50 | 12.53 | 4567.50 | 0.78 | 0.00 |
| 4 | 20 | 6 | 4726.13 | 12.66 | 4726.13 | 1.47 | 0.00 |
| 5 | 25 | 6 | 4842.80 | 12.19 | 4842.80 | 1.55 | 0.00 |
| 6 | 5 | 8 | 3997.50 | 10.76 | 3997.50 | 0.19 | 0.00 |
| 7 | 10 | 8 | 4630.67 | 12.94 | 4630.67 | 1.03 | 0.00 |
| 8 | 15 | 8 | 4793.17 | 13.18 | 4793.17 | 1.30 | 0.00 |
| 9 | 20 | 8 | 4863.02 | 14.21 | 4863.02 | 1.71 | 0.00 |
| 10 | 25 | 8 | 5047.80 | 14.21 | 5047.80 | 2.05 | 0.00 |
| 11 | 5 | 10 | 4166.97 | 11.44 | 4166.97 | 0.24 | 0.00 |
| 12 | 10 | 10 | 4707.54 | 13.31 | 4707.54 | 1.17 | 0.00 |
| 13 | 15 | 10 | 4864.12 | 13.45 | 4864.12 | 1.45 | 0.00 |
| 14 | 20 | 10 | 4981.47 | 14.16 | 4981.47 | 2.08 | 0.00 |
| 15 | 25 | 10 | 5099.68 | 14.56 | 5099.68 | 2.61 | 0.00 |
| 16 | 5 | 6 | 3700.47 | 11.51 | 3700.47 | 0.13 | 0.00 |
| 17 | 10 | 6 | 4366.59 | 13.13 | 4366.59 | 0.57 | 0.00 |
| 18 | 15 | 6 | 4504.84 | 12.93 | 4504.84 | 0.88 | 0.00 |
| 19 | 20 | 6 | 4600.78 | 13.74 | 4600.78 | 0.89 | 0.00 |
| 20 | 25 | 6 | 4697.32 | 13.88 | 4697.32 | 1.43 | 0.00 |
| 21 | 5 | 8 | 3838.69 | 10.56 | 3838.69 | 0.15 | 0.00 |
| 22 | 10 | 8 | 4560.90 | 13.47 | 4560.90 | 0.75 | 0.00 |
| 23 | 15 | 8 | 4769.70 | 13.89 | 4769.70 | 1.21 | 0.00 |
| 24 | 20 | 8 | 4826.15 | 14.88 | 4826.15 | 1.67 | 0.00 |
| 25 | 25 | 8 | 4944.00 | 14.69 | 4944.00 | 2.05 | 0.00 |
| 26 | 5 | 10 | 4033.33 | 11.52 | 4033.33 | 0.15 | 0.00 |
| 27 | 10 | 10 | 4562.67 | 14.69 | 4562.67 | 0.89 | 0.00 |
| 28 | 15 | 10 | 4781.54 | 15.20 | 4781.54 | 1.60 | 0.00 |
| 29 | 20 | 10 | 4959.79 | 14.43 | 4959.79 | 2..06 | 0.00 |
| 30 | 25 | 10 | 5014.05 | 15.24 | 5014.05 | 2.31 | 0.00 |
| Average | | | 4588.54 | 13.18 | 4588.54 | 1.10 | 0.00 |

**Table 6.** Compare with DPC on the medium size 1800 new instances (seed=3)

| instances | *m* | $\varpi$ | DPC | | IHDP | | |
|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | 14974.25 | 277.85 | **14972.15** | 4.51 | **0.000** |
| 2 | 10 | 6 | 17113.60 | 290.08 | 17113.60 | 56.95 | 0.000 |
| 3 | 15 | 6 | 18259.88 | 278.21 | 18259.88 | 112.63 | 0.000 |
| 4 | 20 | 6 | 18250.43 | 310.59 | 18250.43 | 138.08 | 0.000 |
| 5 | 25 | 6 | 18769.08 | 282.40 | 18769.08 | 155.32 | 0.000 |
| 6 | 5 | 8 | 15993.28 | 293.52 | 15993.28 | 8.45 | 0.000 |
| 7 | 10 | 8 | 18022.70 | 283.77 | 18022.70 | 87.99 | 0.000 |
| 8 | 15 | 8 | 18772.30 | 274.74 | 18772.30 | 143.05 | 0.000 |
| 9 | 20 | 8 | 19341.68 | 281.89 | 19341.68 | 192.06 | 0.000 |
| 10 | 25 | 8 | 19653.10 | 287.14 | 19653.10 | 227.91 | 0.000 |
| 11 | 5 | 10 | 15902.78 | 288.95 | 15902.78 | 7.68 | 0.000 |
| 12 | 10 | 10 | 18600.38 | 275.67 | 18600.38 | 99.46 | 0.000 |
| 13 | 15 | 10 | 19213.53 | 275.63 | **19213.**10 | 167.44 | **0.000** |
| 14 | 20 | 10 | 19730.58 | 272.70 | 19730.58 | 207.27 | 0.000 |
| 15 | 25 | 10 | 20012.65 | 282.42 | 20012.65 | 260.77 | 0.000 |
| 16 | 5 | 6 | 14730.78 | 324.04 | **14623.88** | 5.34 | **0.007** |
| 17 | 10 | 6 | 17040.13 | 301.38 | 17040.13 | 47.19 | 0.000 |
| 18 | 15 | 6 | 18025.05 | 305.29 | 18025.05 | 103.17 | 0.000 |
| 19 | 20 | 6 | 18465.53 | 316.33 | 18465.53 | 153.18 | 0.000 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 20 | 25 | 6 | 18842.85 | 313.16 | 18842.85 | 189.40 | 0.000 |
| 21 | 5 | 8 | 15830.78 | 304.80 | **15827.15** | 6.76 | 0.000 |
| 22 | 10 | 8 | 17801.28 | 317.94 | 17801.28 | 81.35 | 0.000 |
| 23 | 15 | 8 | 18732.18 | 299.69 | 18732.18 | 147.56 | 0.000 |
| 24 | 20 | 8 | 19156.50 | 313.54 | 19156.50 | 186.86 | 0.000 |
| 25 | 25 | 8 | 19384.78 | 306.00 | 19384.78 | 216.96 | 0.000 |
| 26 | 5 | 10 | 15078.53 | 297.44 | 15078.53 | 6.96 | 0.000 |
| 27 | 10 | 10 | 17974.48 | 313.68 | **17958.15** | 90.66 | **0.001** |
| 28 | 15 | 10 | 18850.90 | 310.60 | 18850.90 | 166.48 | 0.000 |
| 29 | 20 | 10 | 19401.13 | 311.13 | 19401.13 | 232.41 | 0.000 |
| 30 | 25 | 10 | 20059.15 | 308.62 | 20059.15 | 274.27 | 0.000 |
| Average | | | 18066.14 | 296.64 | 18061.83 | 125.94 | 0.000 |

Note: Bold type in the table represents the non-optimal solution

**Table 7.** Compare with DPC on the small size 1800 new instances (seed=7)

| *Ins.* | *m* | $\varpi$ | DPC | | IHDP | | |
|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | 3947.23 | 2.17 | 3947.23 | 0.12 | 0.00 |
| 2 | 10 | 6 | 4388.65 | 3.11 | 4388.65 | 0.85 | 0.00 |
| 3 | 15 | 6 | 4526.98 | 3.57 | 4526.98 | 1.75 | 0.00 |
| 4 | 20 | 6 | 4654.13 | 3.89 | 4654.13 | 2.23 | 0.00 |
| 5 | 25 | 6 | 4758.02 | 4.21 | 4758.02 | 2.31 | 0.00 |
| 6 | 5 | 8 | 4020.60 | 2.42 | 4020.60 | 0.24 | 0.00 |
| 7 | 10 | 8 | 4507.75 | 3.99 | 4507.75 | 1.44 | 0.00 |
| 8 | 15 | 8 | 4732.47 | 4.66 | 4732.47 | 3.27 | 0.00 |
| 9 | 20 | 8 | 4854.50 | 5.15 | 4854.50 | 3.62 | 0.00 |
| 10 | 25 | 8 | 4924.37 | 5.41 | 4924.37 | 4.56 | 0.00 |
| 11 | 5 | 10 | 4105.08 | 3.45 | 4105.08 | 0.93 | 0.00 |
| 12 | 10 | 10 | 4732.95 | 5.45 | 4732.95 | 3.31 | 0.00 |
| 13 | 15 | 10 | 4899.15 | 6.25 | 4899.15 | 5.70 | 0.00 |
| 14 | 20 | 10 | 5006.60 | 6.72 | 5006.60 | 5.69 | 0.00 |
| 15 | 25 | 10 | 5063.02 | 6.83 | 5063.02 | 7.38 | 0.00 |
| 16 | 5 | 6 | 3810.30 | 4.33 | **3808.07** | 0.20 | 0.06 |
| 17 | 10 | 6 | 4303.37 | 5.14 | 4303.37 | 0.63 | 0.00 |
| 18 | 15 | 6 | 4529.53 | 5.56 | 4529.53 | 0.97 | 0.00 |
| 19 | 20 | 6 | 4598.10 | 5.92 | 4598.10 | 1.42 | 0.00 |
| 20 | 25 | 6 | 4702.70 | 6.22 | 4702.70 | 2.02 | 0.00 |
| 21 | 5 | 8 | 4041.78 | 4.88 | 4041.78 | 0.74 | 0.00 |
| 22 | 10 | 8 | 4501.80 | 5.97 | 4501.80 | 2.29 | 0.00 |
| 23 | 15 | 8 | 4732.78 | 6.74 | 4732.78 | 2.96 | 0.00 |
| 24 | 20 | 8 | 4803.43 | 6.82 | 4803.43 | 3.55 | 0.00 |
| 25 | 25 | 8 | 4851.45 | 7.02 | 4851.45 | 3.88 | 0.00 |
| 26 | 5 | 10 | 4055.08 | 5.12 | 4055.08 | 0.73 | 0.00 |
| 27 | 10 | 10 | 4730.05 | 6.77 | **4729.92** | 2.95 | 0.003 |
| 28 | 15 | 10 | 4820.05 | 7.01 | 4820.05 | 3.91 | 0.00 |
| 29 | 20 | 10 | 4933.32 | 7.40 | 4933.32 | 5.06 | 0.00 |
| 30 | 25 | 10 | 4992.20 | 8.48 | 4992.20 | 6.52 | 0.00 |
| Average | | | 4584.25 | 5.36 | 4584.17 | 2.71 | 0.002 |

Note: Bold type in the table represents the non-optimal solution

**Table 8**. Compare with DPC on the small size 1800 new instances (seed=11)

| *Ins.* | *m* | $\varpi$ | DPC | | IHDP | | |
|---|---|---|---|---|---|---|---|
| | | | OFV | time(s) | OFV | time(s) | GAP(%) |
| 1 | 5 | 6 | 3854.90 | 2.34 | 3854.90 | 0.18 | 0.00 |
| 2 | 10 | 6 | 4352.48 | 3.46 | 4352.48 | 0.90 | 0.00 |
| 3 | 15 | 6 | 4548.65 | 4.18 | 4555.65 | 1.36 | 0.00 |
| 4 | 20 | 6 | 4658.13 | 4.49 | 4658.13 | 1.87 | 0.00 |
| 5 | 25 | 6 | 4761.37 | 4.77 | 4761.37 | 2.43 | 0.00 |
| 6 | 5 | 8 | 3918.62 | 2.87 | 3918.62 | 0.34 | 0.00 |
| 7 | 10 | 8 | 4538.45 | 4.77 | 4538.45 | 1.53 | 0.00 |
| 8 | 15 | 8 | 4758.63 | 5.67 | 4765.73 | 2.55 | 0.00 |
| 9 | 20 | 8 | 4798.27 | 5.45 | 4798.27 | 3.02 | 0.00 |
| 10 | 25 | 8 | 4926.80 | 6.39 | 4926.80 | 3.99 | 0.00 |
| 11 | 5 | 10 | 3963.83 | 3.34 | 3963.83 | 0.69 | 0.00 |
| 12 | 10 | 10 | 4635.02 | 5.56 | 4635.02 | 2.33 | 0.00 |
| 13 | 15 | 10 | 4851.70 | 6.54 | 4851.70 | 4.07 | 0.00 |
| 14 | 20 | 10 | 4926.92 | 6.91 | 4926.92 | 4.80 | 0.00 |
| 15 | 25 | 10 | 5016.78 | 7.57 | 5016.78 | 6.01 | 0.00 |
| 16 | 5 | 6 | 3818.85 | 2.97 | 3818.85 | 0.15 | 0.00 |
| 17 | 10 | 6 | 4257.70 | 3.89 | 4257.70 | 0.61 | 0.00 |
| 18 | 15 | 6 | 4586.68 | 5.02 | 4586.68 | 1.46 | 0.00 |
| 19 | 20 | 6 | 4649.18 | 5.39 | 4649.18 | 2.09 | 0.00 |
| 20 | 25 | 6 | 4776.97 | 5.86 | 4776.97 | 2.43 | 0.00 |
| 21 | 5 | 8 | 3955.52 | 3.69 | 3955.52 | 0.33 | 0.00 |
| 22 | 10 | 8 | 4524.15 | 5.00 | 4524.15 | 1.44 | 0.00 |
| 23 | 15 | 8 | 4716.22 | 6.12 | 4716.22 | 2.62 | 0.00 |
| 24 | 20 | 8 | 4795.03 | 6.08 | 4795.03 | 3.41 | 0.00 |
| 25 | 25 | 8 | 4884.32 | 6.81 | 4884.32 | 4.28 | 0.00 |
| 26 | 5 | 10 | 4100.58 | 4.31 | 4100.58 | 0.58 | 0.00 |
| 27 | 10 | 10 | 4610.17 | 5.85 | 4610.17 | 2.21 | 0.00 |
| 28 | 15 | 10 | 4806.98 | 6.36 | 4806.98 | 3.37 | 0.00 |
| 29 | 20 | 10 | 4908.53 | 7.38 | 4908.53 | 4.58 | 0.00 |
| 30 | 25 | 10 | 4982.80 | 8.07 | 4982.80 | 5.12 | 0.00 |
| Average | | | 4562.81 | 5.24 | 4562.81 | 2.36 | 0.00 |

**Acknowledgements.** This work was supported by the National Natural Science Foundation of China (Grant Nos.61862027, 61702238 and 61866014), the Natural Science Foundation Project of Jiangxi (Grant No.20192BAB207008), the Science Foundation of Educational Commission of Jiangxi Province (Grant Nos.Gjj170316 and Gjj180264).

## References

1. Wäscher G, Haußner H, Schumann H. An improved typology of cutting and packing problems [J]. European Journal of Operational Research, 2007, 183(3): 1109-1130.
2. Wang L, Liu Q, Chen X. Heuristic search algorithm for the rectangular fixed-size guillotine bin packing problem [J]. Journal of Software, 2017, 28: 1640-1654.

3. Song X, Chu C B, Lewis R, et al. A worest case analysis of a dynamic programming-based heuristic algorithm for 2D unconstrained guillotine cutting [J]. European Journal of Operational Research, 2010, 202(2): 368-378.
4. Wuttke D A, Heese H S. Two-dimensional cutting stock problem with sequence dependent setup times[J]. European Journal of Operational Research, 2017, 265: 303-315.
5. Yoon K, Ahn S, Kang M. An improved best-first branch-and-bound algorithm for constrained two-dimensional guillotine cutting problems [J]. International Journal of Production Research, 2013, 51(6): 1680-1693.
6. Herz, J. C. Recursive computational procedure for two-dimensional stock cutting [J]. IBM Journal of Research and Development, 1972, 16(5): 462-469.
7. Beasley, J. E. Algorithms for unconstrained two-dimensional guillotine cutting [J]. Journal of the Operational Research Society, 1985(a), 36(4): 297-306.
8. Carnieri C, Mendoza G A, Luppold W G. Optimal cutting of dimension parts from lumber with a defect: A heuristic solution procedure [J]. Forest Products Journal, 1993, 43: 66-72.
9. Vianna ACG, Arenales MN. Problema de corte de placas defeituosas. Pesqui Operacional, 2006, 26: 185-202.
10. Neidlein V, Vianna, A C G, Arenales, M.N, Wäscher, G. The two-dimensional guillotine-layout cutting problem with a single defect - An and/or-graph approach [J]. Operations Research Proceedings 2008 (Eds.: Fleischmann, B. et al.). Berlin, Heidelberg, Springer-Verlag, 85-90.
11. Afsharian M, Niknejad A, Wäscher G. A heuristic, dynamic programming-based approach for a two-dimensional cutting problem with defects [J]. OR Spectrum, 2014, 36(4):971-999.
12. Martin M, Hokama Pedro H D B, Morabito R, Munari, P. The constrained two-dimensional guillotine cutting problem with defects: an ILP formulation, a Benders decomposition and a CP-based algorithm[J]. International Journal of Production Research, 2019, 1-18.
13. Gonçalves J F, Wäscher G. A MIP model and a biased random-key genetic algorithm based approach for a two-dimensional cutting problem with defects. European Journal of Operational Research, 2020, preprint. doi: https://doi.org/10.1016/j.ejor.2020.04.028
14. Birgin E G, Romão, O. C, Ronconi D P. The multiperiod two-dimensional non-guillotine cutting stock problem with usable leftovers[J]. International Transactions in Operational Research, 2019, 1392-1418.
15. Velasco A S, Eduardo U. Improved state space relaxation for constrained two-dimensional guillotine cutting problems[J]. European Journal of Operational Research, 2019, 272: 106-120.
16. Russo M, Boccia M, Sforza A, Sterle C. Constrained two-dimensional guillotine cutting problem: upper-bound review and categorization. International Transactions in Operation Research, 2020, 27: 794-834.
17. Wu K, Min X, Zhang D. Research on two-dimensional cutting problem with defects. 2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China: 506-511.
18. Neidlein V, Wäscher G. SLOPPGEN: a problem generator for the two-dimensional rectangular single large object placement problem. International Transactions in Operational Research, 2016, 23:173-186.

**Aihua YIN** is a Senior Researcher at Jiangxi University of Finance and Economics. He received the Ph.D. degree in Computer Science from the Huazhong University of Science and Technology, Wuhan, China. His research interests include job shop scheduling problem, cutting and packing problem.

**Chong CHEN** is a Master degree candidate at Jiangxi University of Finance and Economics. He received the Bachelor degree in Computer Science and Technology from the Wuhan University of Science and Technology, China. His research interests include cutting problem.

**Dongping HU** is an associate professor at Jiangxi University of Finance and Economics. She received the Ph.D. degree in Information Security from the Huazhong University of Science and Technology, Wuhan, China. Her research focuses on the searchable encryption, the application intelligent algorithm in information security.

**Jianghai HUANG** is a Master degree candidate at Jiangxi University of Finance and Economics. He received the Bachelor degree in Software Engineering from the Tianjin University Renai College, Tianjin, China. His research interests include cutting problem.

**Fan Yang** is a Master degree candidate at Jiangxi University of Finance and Economics. She received the Bachelor degree in Electronic Commence from the Wuhan University of Science and Technology, Wuhan, China. Her research interests include Text Big Data Analysis.

*Received: January 25, 2020; Accepted: July 02, 2020*

Computer Science and Information Systems 17(3):737–758 https://doi.org/10.2298/CSIS200205018M

# Energy-Efficient Non-linear K-barrier Coverage in Mobile Sensor Network *

Zijing Ma[1], Shuangjuan Li[1] , Longkun Guo[2], and Guohua Wang[1]

[1] College of Mathematics and Informatics, South China Agricultural University, China
mazijingscau@hotmail.com,lishj2013@hotmail.com,w.guohuascut@gmail.com
[2] School of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
forkun@mail.ustc.edu.cn

**Abstract.** K-barrier coverage is an important coverage model for achieving robust barrier coverage in wireless sensor networks. After initial random sensor deployment, k-barrier coverage can be achieved by moving mobile sensors to form k barriers consisting of k sensor chains crossing the region. In mobile sensor network, it is challenging to reduce the moving distances of mobile sensors to prolong the network lifetime. Existing work mostly focused on forming linear barriers, that is the final positions of sensors are on a straight line, which resulted in large redundant movements. However, the moving cost of sensors can be further reduced if non-linear barriers are allowed, which means that sensors' final positions need not be on a straight line. In this paper, we propose two algorithms of forming non-linear k barriers energy-efficiently. The algorithms use a novel model, called horizontal virtual force model, which considers both the euclidean distance and horizontal angle between two sensors. Then we propose two barrier forming algorithms. To construct a barrier, one algorithm always chooses the mobile sensor chain with the largest horizontal virtual force and then flattens it, called sensor chain algorithm. The other chooses the mobile sensor with the largest horizontal virtual force to construct the barrier, other than the mobile sensor chain, called single sensor algorithm. Simulation results show that the algorithms significantly reduce the movements of mobile sensors compared to a linear k-barrier coverage algorithm. Besides, the sensor chain algorithm outperforms the single sensor algorithm when the sensor density becomes higher.

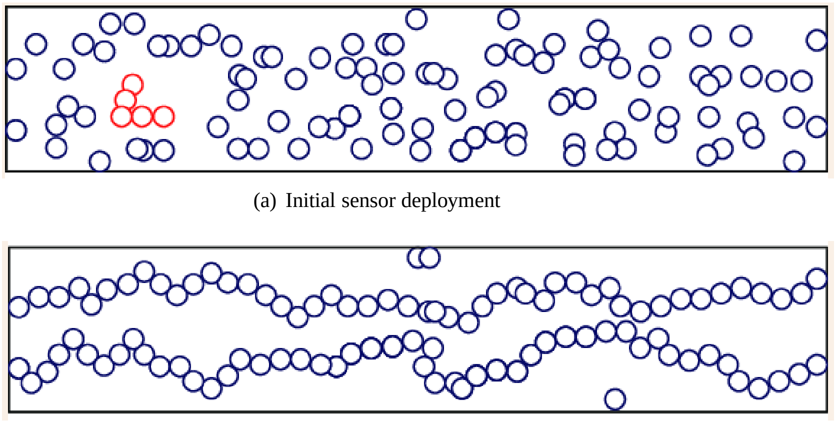
**Keywords:** wireless sensor networks, k-barrier coverage, virtual force, non-linear barrier.

## 1. Introduction

Wireless Sensor Networks(WSNs) have been widely applied in many fields such as intrusion detection, border protection, and environment monitoring. Nowadays, many problems in WSNs have been widely studied such as topology control, localization technology, data aggregation, and coverage problem. Among them, coverage problem is a significant problem in WSNs, which can be classified into many different coverages, including area coverage, target coverage, barrier coverage, and sweep coverage. Barrier coverage is an

* It is an extended version of the ($\mathbf{10}_{th}$ International Symposium on Parallel Architectures, Algorithms and Programming (PAAP 2019))

important problem in these coverage problems. It was first proposed in the work[9] and often used to detect intruders by forming a sensor chain in a belt region of interest(ROI) so that any intruder will be detected when passing through the ROI vertically along any paths. K-barrier coverage is a kind of robust barrier coverage, which guarantees that any intruder crossing the region will be detected by at least k sensors. Initially, sensors are often deployed randomly in the ROI. However, it is not likely to form k-barrier coverage after initial random sensor deployment, as shown in Figure 1(a). With the development of mobile sensor technology, k-barrier coverage can be achieved by moving mobile sensors to the desired positions, as shown in Figure 1(b).



(a) Initial sensor deployment

(b) Final sensor deployment

**Fig. 1.** Initial sensor deployment and final sensor deployment

However, mobile sensor is equipped with batteries, and it costs much more energy during the sensor movement than the sensing. Thus, it is important to minimize the sensor movements for prolonging the network lifetime while achieving k-barrier coverage. Some algorithms [13, 10] are proposed to form linear barriers using mobile sensors, which means that the sensors move to locate on a line segment spanning the region. Obviously, it results in large redundant sensor movements by mobile sensors to form linear barriers. To further reduce the sensor movements, the work in [1, 16, 12] formed a non-linear barrier energy-efficiently, which means the sensors' final positions are on a curve, other than a straight line, as shown in Figure 1(b). It was showed that the sensor movements can be reduced for forming a non-linear barrier than a linear barrier. However, very few works studied how to form k non-linear barriers energy-efficiently, which is a challenging problem. This paper tries to propose solutions to solve this problem.

The work in [1] proposed an algorithm to form a non-linear barrier coverage energy-efficiently, which outperformed other existing algorithms. However, this algorithm cannot be extended to the case of k-barrier coverage directly. Inspired by the work[1], we propose two energy-efficient algorithms of forming non-linear k barriers based on the initial deployment of mobile sensors. In the work[1], virtual force model has been proposed to pull one sensor chain to touch another sensor chain. This traditional virtual force only considers the euclidean distance of two sensors, which might cause part of barriers formed vertically. Instead, we define the notion of horizontal virtual force by considering the euclidean distance and also horizontal angle. In this paper, we propose two solutions for achieving non-linear k-barrier coverage based on the horizontal virtual force, called sensor chain algorithm and single sensor algorithm respectively. The main idea of the two solutions are to first divide the region into several subregions, and then construct k sensor chains, called sub-barriers, crossing from the left boundary of each subregion to the right boundary respectively, and finally connect the sub-barriers in neighbor subregions for forming k barriers in the whole region. We use two different sub-barrier forming algorithms in these solutions. One algorithm always chooses the mobile sensor chain with the largest horizontal virtual force and then flattens it, called mobile sensor chain movement algorithm. The other algorithm always chooses the mobile sensor with the largest horizontal virtual force to construct the sub-barrier, other than the mobile sensor chain, called single sensor movement algorithm. Simulation results show that our proposed algorithms efficiently decrease the movements of mobile sensors compared to a linear k-barrier coverage algorithm. The simulations also show that the sensor chain algorithm outperforms the single sensor algorithm when the sensor density becomes higher.

In summary, the main contributions of this paper are listed as follows:

- We study the problem of forming non-linear k-barrier coverage using mobile sensors and propose two energy-efficient solutions.
- We introduce the horizontal virtual force model. It considers both euclidean distance and horizontal angle between two sensors, which can avoid forming part of the barriers vertically.
- We first divide the region into several subregions, and then construct k sensor chains, called sub-barriers, crossing from the left boundary of each subregion to the right boundary respectively, and finally connect the sub-barriers in neighbor subregions for forming k barriers in the whole region.
- We propose two algorithms to efficiently form k sub-barriers in each subregion.
- Simulation results demonstrate the efficiency of our proposed algorithms.

The rest of paper is organized as follows. Section 2 reviews some related work about barrier coverage using mobile sensors. Section 3 establishes the networks model and gives some terms about our algorithms. Section 4 describes an algorithm of forming one sub-barrier in a subregion called mobile sensor chain movement algorithm. Section 5 proposes an algorithm of forming one sub-barrier in a subregion called single sensor movement algorithm. In Section 6 we propose the solutions of forming k barriers in the whole region. In Section 7 we evaluate the performance of the algorithms. Section 8 concludes our paper.

## 2. Related Work

Barrier coverage has been widely studied in wireless sensor networks. The notion of k-barrier coverage was first proposed by the work[9]. In the work [9], two kinds of barrier coverage were proposed: weak barrier coverage and strong barrier coverage. Weak barrier coverage aims at detecting those intruders which cross the ROI along the vertical paths, while the strong barrier coverage aims to detect any intruder crossing the ROI along any paths. In this paper, we aim to form strong barrier coverage. Existing barrier coverage algorithms can be divided into centralized algorithms and distributed algorithms.

The centralized algorithms are under the assumption that all the information of region and the locations of mobile sensors are known beforehand. A central device can compute the final location of each mobile sensor and then sensors can move directly to their final location. The work in [2] constructed linear k barriers by dividing the region into subregions and forming a baseline grid barrier and an isolation grid barrier in each subregion. However, a large number of redundant sensors are needed for constructing isolation grid barriers. Especially when there are not enough sensors remained in one subregion to fill the isolation grid, the algorithm has to move mobile sensors from other subregions to form the grids, which might cause lots of redundant movements. The work in [7] defined a novel barrier coverage model, called hetebar, and gave a centralized algorithm to find out the maximum lifetime of hetebar after sensors' initial deployment. The work in [13] tried to move the sensors to the grid points while minimizing the maximum sensor movement. The work in [10] proposed a polynomial-time algorithm to move the sensors to cover the barrier line while minimizing the maximum sensor movement, which can reduce the sensor movement. All the above algorithms formed linear barriers, which means the sensors in the barriers must locate on a straight line. The work in [1] proposed an energy-efficient algorithm based on virtual force to form non-linear barrier coverage, which outperformed the linear barrier algorithms in the work[2]. The work in [16] proposed a centralized algorithm to form barriers with mobile sensors under the influence of both sunny and rainy days. The work in [14] studied the hybrid network consisting of stationary sensors and mobile sensors and proposed an algorithm of relocating mobile sensors to improve barrier coverage by filling the gap resulted by stationary sensors. In [19], similar to [14], the authors tried to use mobile sensors and stationary sensors to form barrier coverage. They proposed a two-phase deployment algorithm, where the stationary sensors are first deployed, and then the mobile sensors are deployed to fill the gaps between stationary sensors to form a barrier. Moreover, they proposed a scheme based on probabilistic model to minimize the total sensor cost. The work in [17] achieved barrier coverage in heterogeneous WSNs by leveraging various types of mobile sensors and proposing a greedy movement algorithm to fill the gaps between stationary sensors deployed.

In fact, the performance of centralized algorithms might be limited by the central device if there are a large number of sensors. Hence, some researchers studied the distributed algorithms. Mobile sensor adjusts its location according to its environment and it does not need to know all the information of other sensors. However, these algorithms might cause large redundant movements of mobile sensors in practical applications. The work in [4] proposed an algorithm, which was inspired by the animal aggregations, to solve the problem of establishing barrier coverage between two landmarks. The work in [8] proposed a fully distributed algorithm based on virtual force to relocate the sensors from the original positions to uniformly distribute on the convex hull of the region. The work in [15] presented a distributed algorithm called MobiBar to form linear k-barrier coverage. Furthermore, the authors proved the algorithm terminated in a finite time. The work in [12] proposed two distributed algorithms for forming barriers based on virtual force. However, it was not energy-efficienct. The work in [3] studied the sensors' movement in barrier coverage with a game theory approach. In [18], the authors formed barrier coverage using directional sensors in a line-based model. The algorithm indicated that after deployed along a predetermined line, the directional sensor can rotate itself based on the information of its adjacent sensors to form barrier coverage.

## 3. Network Model

Assuming that there is a rectangular belt region of length *L* and width *H*, which is *L*≫*H*. A set of mobile sensors is deployed randomly in this region. The sensing range of these sensors is $R_s$ and the number of the sensors is *N*. The initial position of mobile sensor $s_i$ is $(x_i, y_i)$. We assume that each mobile sensor knows its coordinate using GPS system. Each sensor can move in all direction, whose moving distance is the euclidean distance of its initial position and final position.

In this paper, we focus on how to form k-barrier coverage energy-efficiently using mobile sensors. K barriers are formed by k chains of sensors whose sensing range overlap with each other crossing from the left boundary of the region to the right boundary. We study how to find the sensors' final positions so that the sensors can move to form k barriers crossing the region while minimizing the average sensor moving distances.

Before showing the algorithm, we will define some terms below.

**Definition 1.** ***Mobile Sensor Chain****: A mobile sensor chain is a set of mobile sensors in which the sensing range of each sensor should intersect with that of adjacent sensor in this set, which means the distance between these two mobile sensors is less than or equal to* $2R_s$.

A mobile sensor chain, denoted in red, can be seen in Figure 1(a). Note that a single sensor is the minimum mobile sensor chain. A barrier is formed when there is a mobile sensor chain in which there are two mobile sensors whose sensing ranges intersect the left boundary and the right boundary respectively.

**Definition 2.** ***Main Mobile Sensor Chain****: A main mobile sensor chain is a kind of mobile sensor chain in which the sensing range of one sensor intersects with the left boundary.*

We form a barrier by constructing a main mobile sensor chain from the left boundary to the right boundary of the region.

Now we'll define the notion of horizontal virtual force by considering the euclidean distance and also horizontal angle.

Let $N$ represent the set of all sensors deployed in the region and $N_c$ represent all the mobile sensors in the main mobile sensor chain. For each mobile sensor $c \in N_c$ and $v \in N - N_c$, we define horizontal virtual force $h(c, v)$ from $v$ to $c$ as follows:

$$h(c, v) = \frac{\alpha}{distance(c, v)} \times cos\theta \tag{1}$$

$$c \in N_c, v \in N - N_c, \theta \in (0, \frac{\pi}{2})$$

In the formula, $distance(c, v)$ is the Euclidean distance between mobile sensor $c$ and mobile sensor $v$. The direction of $h(c, v)$ starts from $v$ and points to $c$. $\theta$ is the included angle of the horizontal line and the line of mobile sensor $c$ and $v$ . Note that $\theta \in (0, \frac{\pi}{2})$ and $\alpha$ is a scaling parameter.

For each sensor $c \in N_c$, we can calculate the maximum horizontal virtual force $H\left(c\right)$ as follows:

$$H(c) = \underset{v}{max} h(c, v) \tag{2}$$

$$c \in N_c, v \in N - N_c, \theta \in (0, \frac{\pi}{2})$$

The action point is defined as the sensor $c_m \in N_c$ which satisfies that $H\left(c_m\right)$=$\underset{c}{max} H(c)$, $\forall c \in N_c$. The reaction point is defined as the sensor $v_m \in N - N_c$ whose horizontal virtual force to sensor $c_m$ is $H\left(c_m\right)$.

## 4. Mobile Sensor Chain Movement Algorithm

In this paper, the main idea of forming k barriers is to first divide the region into several subregions, then construct k sub-barriers from the left boundary of each subregion to the right boundary and finally connect the sub-barriers in adjacent subregions for forming k barriers in the whole region.

In this section, we will show an energy-efficient algorithm of forming a sub-barrier called mobile sensor chain movement algorithm. The whole solution of forming k barriers will be discussed in section 6.

The main idea of mobile sensor chain movement algorithm is to always choose the mobile sensor chain with the largest horizontal virtual force to construct the sub-barrier and then flatten it. This algorithm can be described in three phases.

In the first phase, called the left-fix phase, we will find out the leftmost mobile sensor and then pull it to the left boundary as well as the chain where the mobile sensor belongs to. This chain is regarded as the main mobile sensor chain.

In the second phase, called the extending phase, we will select the rightmost sensor of this main mobile sensor chain and compute the sensor chain with the largest horizontal virtual force towards the rightmost sensor, and pull the mobile sensor chain towards the main mobile sensor chain until the mobile sensor chain touches the rightmost sensor.

In the third phase, called the right-fix phase, when the main mobile sensor chain touches the right boundary of the subregion, we will select the corresponding main mobile sensor chain in the next subregion and move sensors to fill the gap between the chains. If it is the last subregion, then this phase will be ignored. Once the main mobile sensor chain touches its corresponding chain or the right boundary of subregion, it implies that one sub-barrier is formed.

### 4.1. Flattening Algorithm

Before showing the detail of these three phases, we first present the flattening algorithm.

The flattening algorithm is used when a mobile sensor chain is selected to move to the target which may be the left boundary of the region or a main mobile sensor chain. The main idea of this flattening algorithm is that we first compute the horizontal path of this mobile sensor chain and then pull the sensors in this chain toward the target one by one by extending the horizontal path.

The detail of this algorithm is described as in Algorithm 1.

**Algorithm 1** Flattening Algorithm

```
Input: main mobile sensor chain M, mobile sensor chain G
Output: updated main mobile sensor chain M
Find out the rightmost sensor of M, regard it as an action point
Calculate the sensor with the largest horizontal force towards the action point, regard it as a
reaction point
if the reaction point is on the rightside of action point
    Compute the horizontal path H starting from the reaction point.
    for each mobile sensor $c \in H$
        moving sensor = c
        The moving sensor c moves towards action point along the straight line and touches the
action point
        action point = moving mobile sensor
        if the action point intersects with the mobile sensor chain then
            return M
        end if
        if moving sensor's degree $\geq 2$ then
            Compute the redundant sensor chain R which touches the moving sensor
            for each mobile sensor $d \in R$
                moving sensor = d
                The moving sensor d moves towards action point along the straight line and touches
the action point.
                action point = moving sensor
                if the action point intersects with the mobile sensor chain G then
                    return M
                end if
            end for
        end if
    end for
else
    reaction point moves to action point's position
    action point moves towards the right boundary of subregion until it is tangent with the reac-
tion point
return M
```

Let $N$ denote all mobile sensors and $N_a$ denote those mobile sensors in the main mobile sensor chain.

First, we enumerate all the mobile sensors in the main mobile sensor chain and find out the sensor with rightmost X coordinate. We regard this sensor as the action point $a$. For each mobile sensor $r \in N - N_a$, compute all the horizontal virtual force from $a$ to $r$ and we need to find out the largest horizontal virtual force among them. The sensor with the largest horizontal virtual force is the reaction point. Let $P$ denote the mobile sensor chain where the reaction point is. We'll extend the main mobile sensor chain by pulling $P$ towards it.

Second, compute the horizontal path of $P$. we enumerate all possible paths starting from this reaction point by doing a depth-first search and calculate the variance of Y coordinate of these paths and select the path with the lowest variance.

Third, if the reaction point is on the rightside of $a$, for each sensor $c$ in the horizontal path, move $c$ towards the main mobile sensor chain by extending the horizontal path $P$. We first compute the degree of the reaction point and then move the reaction point to touch the action point of the main mobile sensor chain. Then the reaction point becomes the new action point. If the reaction point's degree is greater than or equal to 2, it means there is at least one redundant mobile sensor chain $R_G$ connecting the reaction point before it moves. Therefore, moving $R_G$ is preferential and we will move the sensors in $R_G$ towards the action point along the straight line one by one until the new action point intersects with the mobile sensor chain or all the mobile sensors in $R_G$ have been moved. If the new action point intersects with $P$, the procedure stops; If all mobile sensors in $R_G$ have been moved, then we will continue moving those mobile sensors in the horizontal path $P$.

However, if the reaction point is on the left of $a$, we need to insert the reaction point of the mobile sensor chain into the main mobile sensor chain. First we move the reaction point to the position of $a$. Then, to extend the main mobile sensor chain, we pull $a$ towards the right boundary of the subregion until $a$ is tangent with the reaction point.

To illustrate the algorithm more clearly, we will show two simple examples. Figure 2 shows the example when the reaction point is on the rightside of the action point and figure 3 shows the example when the reaction point is on the left.

Figure 2(a) shows the initial deployment of the main mobile sensor chain and another mobile sensor chain $D_G$. As is shown in the figure, the rightmost mobile sensor in main mobile sensor chain is selected as the action point and the reaction point in $D_G$ is selected as the moving mobile sensor. By traversing the horizontal path, move the moving mobile sensor towards action point along the straight line until keeping their sensing ranges just intersect at one point, which is tangent. Then this moving sensor will become the new action point and the next mobile sensor in horizontal path will be the new reaction point, which is becoming the moving sensor.

In Figure 2(b), we can find that the degree of moving sensor is 2, which means there is a redundant mobile sensor chain touching the moving sensor, so we need to record this redundant mobile sensor chain, and then move the moving sensor.

In Figure 2(c), since we have recorded this redundant mobile sensor chain, we have to move the redundant mobile sensor chain instead of mobile sensors in the horizontal path. For each mobile sensor in the redundant mobile sensor chain, we will move it towards the action point along the straight line and then it become the new action point until all mobile sensors are added to the main mobile sensor chain.

In Figure 2(d), the redundant mobile sensor chain has been moved to touch the main mobile sensor chain. Note that adding the redundant mobile sensor chain increases the horizontal length of the main sensor chain. Next, we keep moving the mobile sensors in horizontal path until there is another redundant mobile sensor chain or no mobile sensors in the horizontal path.

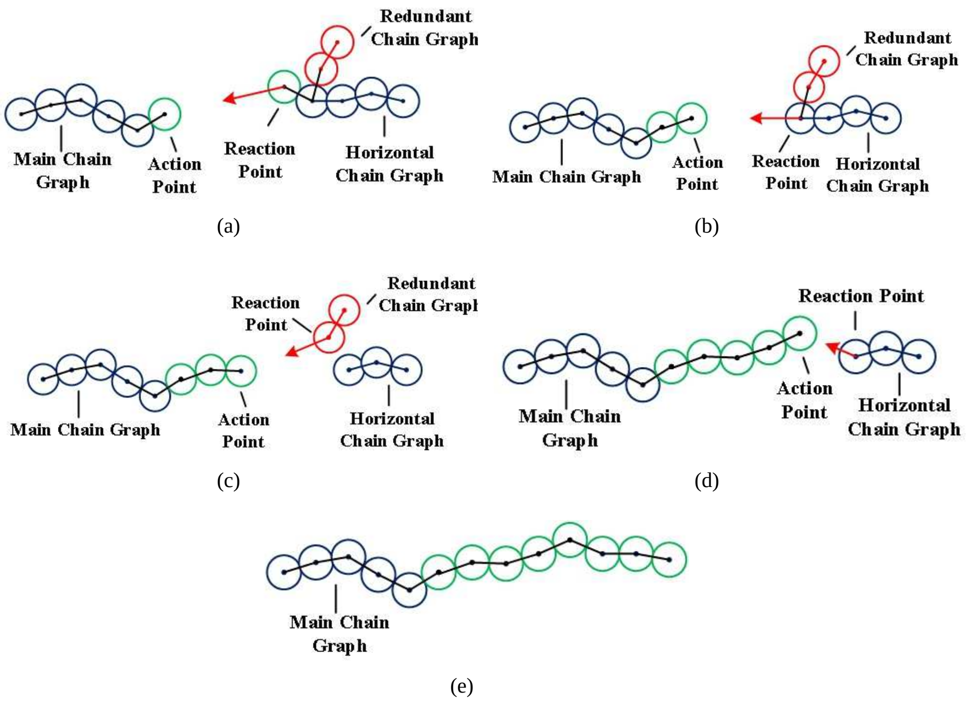


**Fig. 2.** An example of flattening algorithm when *reaction point* is on the right

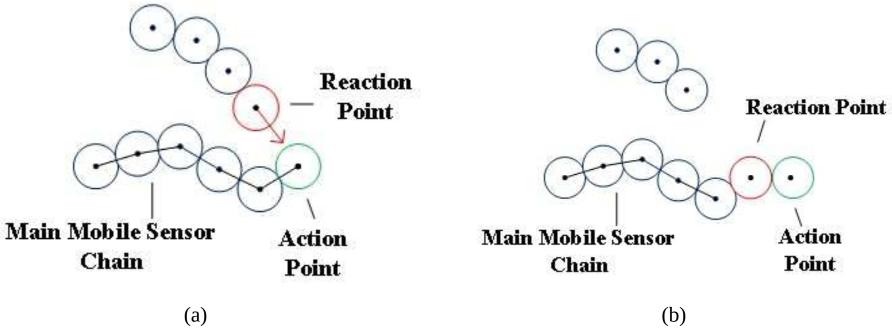


**Fig. 3.** An example of flattening algorithm when *reaction point* is on the left

Figure 2(e) shows the result of the flattening algorithm. We can see the main mobile sensor chain merges the other mobile sensor chain and extends in horizontal direction.

Figure 3 shows an example when the reaction point is on the left of the action point. As we can see, the reaction point moves to the action point's position, then the action point moves towards the right boundary of subregion and finally, it is tangent with the reaction point. Note that when the reaction point is on the left of the action point, we only move the reaction point to its corresponding position, not the mobile sensor chain where the reaction point is.

### 4.2. Forming One Sub-barrier

Now we'll show the mobile sensor chain movement algorithm of forming one sub-barrier. The algorithm is divided into three phases: Left-Fix Phase, Extending Phase and Right-Fix Phase.

In the Left-Fix Phase, we will identify the main mobile sensor chain in the subregion. We find out the leftmost mobile sensor chain, which is the closest mobile sensor chain $L_G$ to the left boundary of the subregion. Next, we will move $L_G$ to the left boundary, making the leftmost mobile sensor in $L_G$ tangent with the left boundary. Note that we move $L_G$ using the flattening algorithm, and the action point of the algorithm is a virtual sensor, and its coordinate is $(-R_s, Y_{leftmost})$. The $Y_{leftmost}$ is the Y coordinate of the leftmost mobile sensor in $L_G$. In the end, we regard $L_G$ as the main mobile sensor chain.

In the Extending Phase, we try to extend the main mobile sensor chain by moving other mobile sensor chains towards it. We will first calculate the reaction point and the action point, where the reaction point is the sensor with the largest horizontal virtual force. Move the mobile sensor chain containing reaction point to the main mobile sensor chain using the flattening algorithm. We continue iterating this phase until the main mobile sensor chain touches the right boundary of subregion or no redundant sensor can be selected. When the phase stops, we come to the right-fix phase.

In the Right-Fix phase, since the main mobile sensor chain has touched the right boundary of subregion, we have to connect it to another main mobile sensor chain in the next subregion to form a barrier.

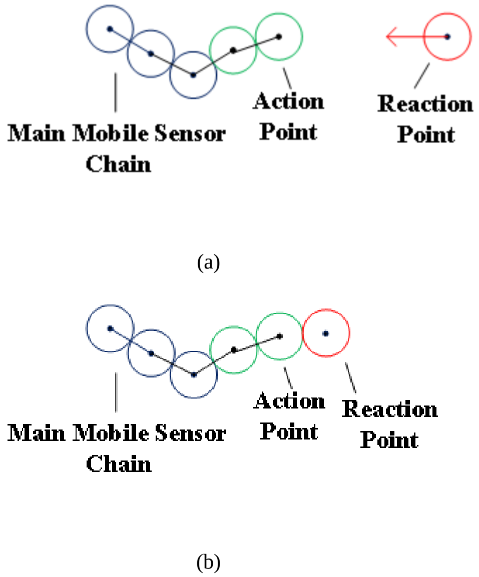
## 5. Single Sensor Movement Algorithm

In section 4, we proposed a mobile sensor chain movement algorithm and the flattening algorithm. However, when the length of ROI is not long, flattening algorithm works not well since there is no space to move the mobile sensor chain. Motivated by this, we will introduce a single sensor movement algorithm.

The main idea of this algorithm is that we first divide the ROI into several subregions and in each subregion we select k mobile sensors as the sub-barriers, also called main mobile sensor chain. Then we extend these sub-barriers by moving the sensors with the largest horizontal virtual force towards the sensors in the main mobile sensor chains until the main mobile sensor chain touches another main mobile sensor chain in the next subregion or the right boundary of the ROI.

### 5.1. Extending Chain Algorithm

In this subsection, we propose an algorithm of forming one sub-barrier. This algorithm can be divided into three phases: Left-Fix Phase, Extending Phase, and Right-Fix Phase.



(a)

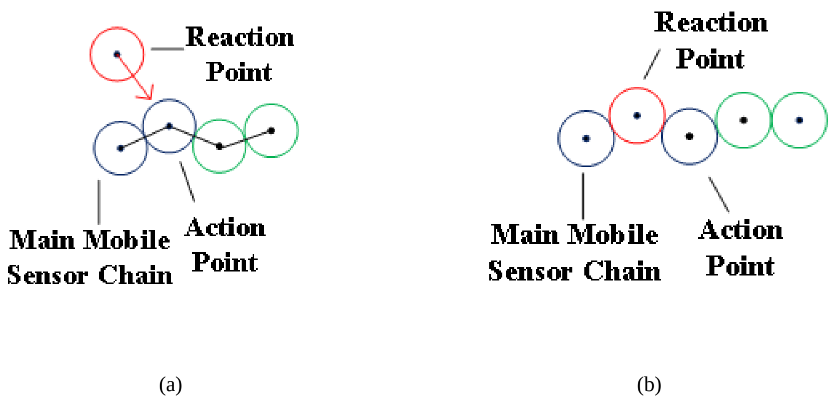
(b)

**Fig. 4.** An example of extending chain algorithm when *reaction point* is on the right

In the Left-Fixed phase, in each subregion, at first, we find out k sensors which are the closest to the left boundary of subregion by calculating the distances between sensors in the subregion and the left boundary of the subregion. Then move these sensors to touch the left boundary of subregion. The final positions of these k sensors should be $(i_{subregion} * L_{subregion} + R_s, Y_{sensor})$, where $i_{subregion}$ is the number of region, starting from 0. $L_{subregion}$ is the length of subregion, $Y_{sensor}$ is the sensors' Y coordinate. Note that we only move these k sensors, which are regarded as the main mobile sensor chain.

In the Extending Phase, we extend the main mobile sensor chain. First, we calculate all the horizontal virtual forces between the sensors in the main mobile sensor chain and other sensors in this subregion, and choose the sensor with the largest horizontal virtual force. Then find out the action point and reaction point. If the reaction point is on the right side of the action point, we just move the reaction point to touch the action point along the line segment between them. But if the reaction point is on the leftside of the action point, we need to move the reaction point to the action point's position, and then, for each sensor on the right side of the action point on the chain, except the rightmost sensor(the last sensor of the chain), it moves to touch its right neighbor sensor. Note that the rightmost sensor does not have the right neighbor sensor, so move it towards the right boundary of subregion.

In the Right-Fixed phase, it is the same as mobile sensor chain movement algorithm.



(a) (b)

**Fig. 5.** An example of extending chain algorithm when *reaction point* is on the left

**Algorithm 2** Extending Chain Algorithm

```
Input: mobile sensor set $S$, main mobile sensor chain $M$
Output: updated main mobile sensor chain $M$
for each sensor $c \in M$
    calculate the horizontal virtual force between $c$ and other sensors which are not in all the main mobile sensor chain
    select the sensor with the largest horizontal force, as the reaction point of $c$
end for
find out the action point whose horizontal virutal force between action point and reaction point is the largest
if the reaction point is on the rightside of action point
    moving sensor = reaction point
    The moving sensor moves towards action point along the straight line and touches the action point
else
    insertPosition = $M$.indexOf(action point)
    reaction point moves to action point's position
    for each sensor $c_{insertPosition} \in M$
        if insertPosition $>=$ $M$.size-1
            $c_{insertPosition}$ moves towards the right boundary of subregion while touching its left neighbor sensor
        else
            $c_{insertPosition}$ moves to $c_{insertPosition+1}$ point's position
        insertPosition = insertPosition + 1
    end for
return $M$
```

The detail of this algorithm is described in Algorithm 2.

Also, to illustrate the algorithm clearly, we present two examples of extending chain algorithm in the extending phase. Figure 4 shows an example of extending chain algorithm when the reaction point is on the right side of the action point. The reaction point moves towards the action point along the line segment between reaction point and the action point until the reaction point touches the action point. Figure 5 shows how the algorithm works when the reaction point is on the leftside of the action point. The reaction point moves to action point's position. Then the action point moves to the position of its right neighbor sensor. Also, this sensors on the rightside of the action point moves to the position of its right neighbor sensor except the rightmost sensor. The rightmost sensor moves towards the right boundary of the subregion while touching its left neighor sensor. Note that the extending chain algorithm is used to extend the length of main mobile sensor chain.

## 6. Forming K-barrier

In this section, we'll show how to form k barriers. Actually, mobile sensors chain movement algorithm and single sensor movement algorithm in the former two section aim to forming one sub-barrier in a subregion. We will propose two solutions of forming k barriers based on these two algorithms, called sensor chain algorithm and single sensor algorithm.

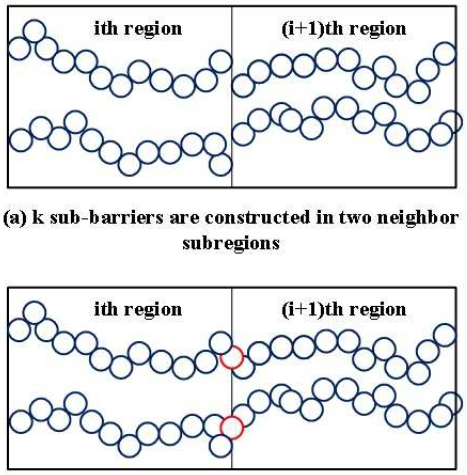
The main idea is to first divide the region into several subregions, and then use the two algorithms k times in each subregion to form k sub-barriers respectively. After forming k sub-barriers, we will connect the k sub-barriers with the other sub-barriers in the right subregions so that the k barriers is formed.

The detail of the solutions are described as follows:

First, the region is divided into equal-sized subregions whose length are $L_r$ and the width are $W$, where $L_r = L/n$ and *n* is the number of subregions. Empirical, *n* should not be too large or too small. On one hand, if *n* is too large, there might be not enough sensors to form barriers in the subregion. On the other hand, if *n* is too small, the length of subregion will be longer, which might result in a larger moving distance of mobile sensors.

Second, form k sub-barriers in each region independently by running the two algorithms of forming one sub-barrier k times. The number of sensors constructing one barrier is limited to be $N_i/k$, avoiding that there is not enough sensor to construct the $k_{th}$ sub-barrier, where $N_i$ is the number of sensors deployed in $i_{th}$ subregion. After completing k sub-barriers in each subregion, it can be observed that these sub-barriers in two adjacent subregions may be not connected, as shown in Figure 6(a).

Third, for simplicity, k sub-barriers in one subregion is called the left k sub-barriers while k sub-barriers in its right neighbor subregion is called the right k sub-barriers. The left or right k sub-barriers are numbered increasingly by their locations on the right or left boundary of their subregion respectively. Each left sub-barrier connects with the same number of the right sub-barrier by pulling the sensors one by one to fill the gap between them. For example, sensors denoted in red are moved to connect sub-barriers in adjacent subregions, as shown in Figure 6.



(a) k sub-barriers are constructed in two neighbor subregions

(b) k sub-barriers are connected in two neighbor subregions

**Fig. 6.** Connecting two sub-barriers in two adjacent subregion

Finally, k barriers will be formed in the whole region. Note that the solutions will form non-linear barriers in the region, so the shapes of barriers are curves. The time complexity of the solutions are $O(n^2)$.

## 7. Simulation Results

In this section, we evaluate the performance of our proposed algorithms using Java, called sensor chain algorithm and single sensor algorithm. These two algorithms are compared with the CBIGB algorithm in the work [2]. The CBIGB algorithm constructs k barriers by dividing the region into equal-sized subregions, selecting a baseline based on the distance between sensors and baselines, and forming a baseline grid barrier and an isolation grid barrier using hungarian algorithm in each subregion. The results obtained are the average of running the experiments 100 times.

Figure 7,8,9 and 10 show how the average moving distance of sensors changes as the number of sensors increases. Sensors are deployed in the regions with length 30m, 50m, 100m, and 150m respectively and width 8m. Sensors' sensing radius is 0.5m. The number of sensors is different according to the length of regions. We divide the region into 3 subregions. It can be observed that our algorithms result in less average moving distances of sensors than CBIGB algorithm in all the figures.

Figure 7 shows the performance of algorithms when sensors are deployed in a 30m×8m region. As the number of sensors increases, our proposed algorithms always obtain a smaller average moving distance than CBIGB algorithm. Meanwhile, the single sensor algorithm performs better than sensor chain algorithm.

Figure 8 shows the performance of our algorithms when the length of the region is 50m. It can be seen that the result obtained by our algorithms is almost half of that by

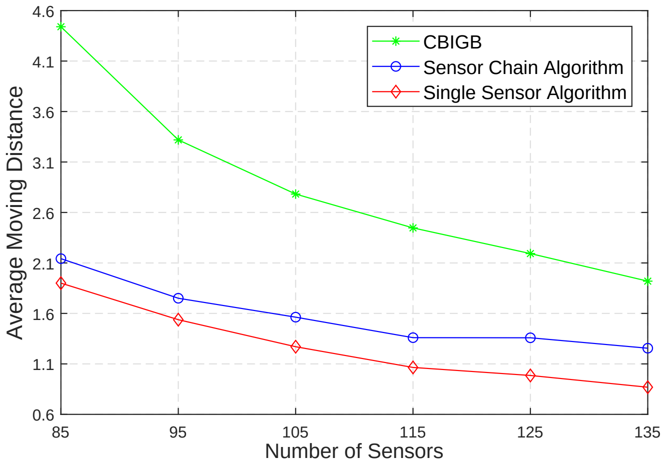


**Fig. 7.** Average moving distance vs number of sensors in 30m×8m ROI

CBIGB algorithm. At first, the sensor chain algorithm performs worse than single sensor algorithm. However, as the number of sensors increases, the results by the sensor chain algorithm improve sharply. When the number of sensors is 170, they achieve almost the same result.

Figure 9 shows the performance of our algorithms when the length of the region is 100m. At first, the sensor chain algorithm results in a larger moving distance than single sensor algorithm. As the number of sensors increases, the moving distances by these two algorithms both decrease. But the result of the sensor chain algorithm decreases more sharply. When the number of sensors is 275, the results of these two algorithms become the same. As the number of sensors increases, the sensor chain algorithm has a smaller result than the second algorithm. It implies in the middle region case, the sensor chain algorithm is more suitable when the number of sensors is large.

In figure 10, sensors are deployed in the large region with length 150m and width 8m. It can be seen that the sensor chain algorithm results in a larger moving distance than single sensor algorithm. However, as the number of sensors increases, the result of the first algorithm decreases sharply and that by the second algorithm is almost the same. Thus, when the number of sensors is small, single sensor algorithm is more suitable.

Figure 7,8,9 and 10 imply that a larger number of sensors leads to less average moving distance. That is because a larger number of sensors means a higher node density, making sensors move less distance. Additionally, our algorithms are more energy-efficient than CBIGB algorithm in different size of regions since our algorithms result less average moving distance. When the number of sensors is small, single sensor algorithm outperforms the sensor chain algorithm. When the number of sensors becomes larger, the sensor chain algorithm is more suitable.

Next, we study the performance of our proposed algorithms and CBIGB algorithm when the sensors' sensing radius varies. Sensors are randomly deployed in a region with length 50m and width 8m. The number of sensors is 220. Figure 11 shows that the average moving distance decreases as the sensors' sensing radius becomes larger, since sensors with larger sensing radius can cover larger areas and thus other sensors can move less to touch it. It is observed that the result of sensor chain algorithm and single sensor algorithm is almost the same. When the sensing radius is 0.3m, the average moving distance is almost 2m. When the sensing radius is 0.55m, the moving distance is almost 1m. Hence, a larger sensing radius is preferable for reducing the moving distance.

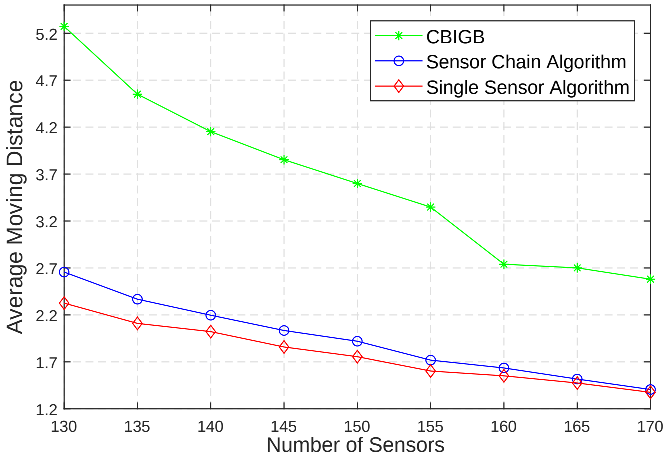


**Fig. 8.** Average moving distance vs number of sensors in 50m×8m ROI

At last, we will evaluate the performance of three algorithms when the length of region varies. Assuming that the node density is a constant value and then we increase the length of region to simulate large region. Note that the node density can be calculated by $N/S$, where $N$ is the number of sensors and $S$ is the area of ROI. Therefore, as the length of region increases, the number of sensors also increases. Sensor's sensing radius is set to be 0.5m and the node density varies from 0.375, 0.4, and 0.425.

In figure 12, the length of region varies from 30m to 100m with step 10m and the number of sensors ranges from 90 to 300 with step 30. As the length of region increases, the result by sensor chain algorithm decreases, while that by single sensor algorithm increases. It implies that the sensor chain algorithm outperforms the single algorithm when the length of the region is long. When the length of the region is 60, they share the same result. The curves in figure 13 and 14 are similar to that of figure 12. The sensors' density is 0.4 and 0.425 and the number of sensors varies from 96 to 320 and 102 to 340 respectively. When the length of region is 60m, these two algorithms obtain the same result.

In summary, figure 12,13, and 14 show that the average moving distance obtained by our algorithm is always less than CBIGB algorithm when the length of region increases, which implies that our algorithms are scalable and can be applied to large scale sensor networks. Moreover, single sensor algorithm outperforms the sensor chain algorithm when

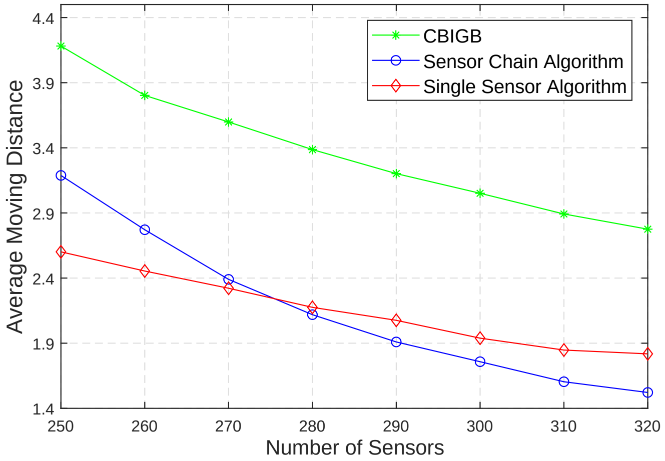


**Fig. 9.** Average moving distance vs number of sensors in 100m×8m ROI

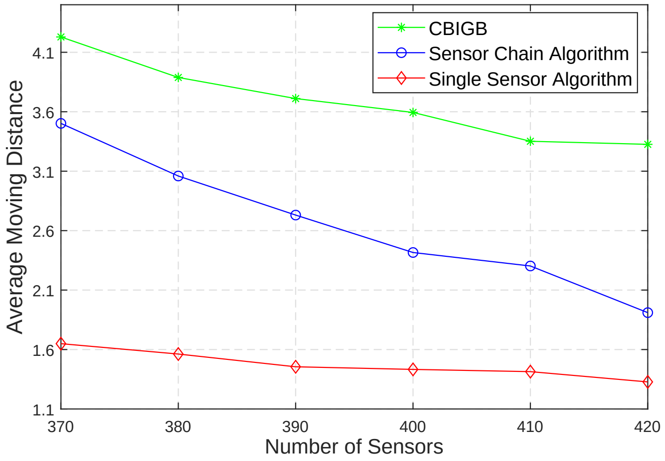


**Fig. 10.** Average moving distance vs number of sensors in 150m×8m ROI

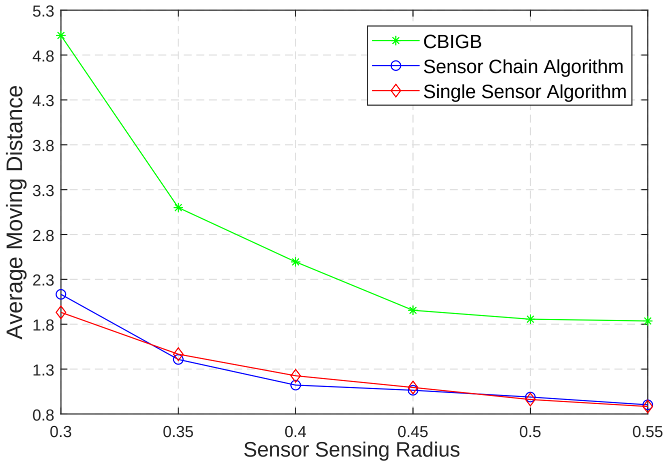


**Fig. 11.** Average moving distance vs sensor sensing radius in 50m×8m ROI

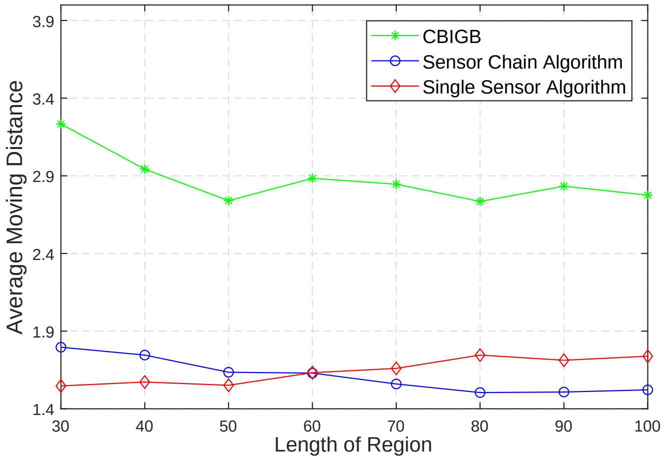


**Fig. 12.** Average moving distance vs length of region with width 8m and sensors' density 0.375

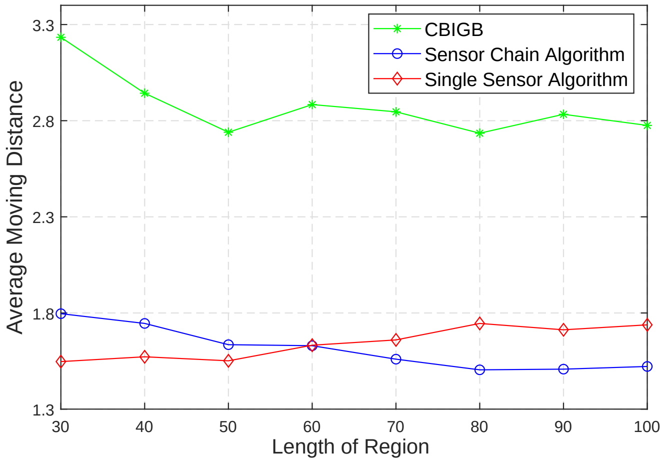


**Fig. 13.** Average moving distance vs length of region with width 8m and sensors' density 0.4

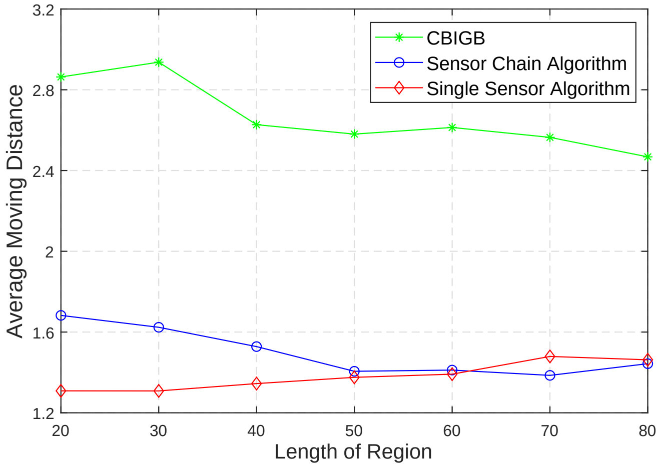


**Fig. 14.** Average moving distance vs length of region with width 8m and sensors' density 0.425

the length of region is short. However, when the length of region becomes longer, the sensor chain algorithm is more suitable.

Our proposed algorithms outperform CBIGB algorithm in average moving distance due to two reasons. First, CBIGB algorithm forms linear k-barrier coverage by moving sensors to some predetermined baselines, which causes large redundant movements. Second, besides forming sub-barriers in subregions, CBIGB algorithm has to form isolation grid barrier vertically to combine sub-barriers between adjacent subregions. Obviously, it also increases the movements of sensors.

## 8. Conclusion

In this paper, we propose two algorithms based on horizontal virtual force model, called sensor chain algorithm and single sensor algorithm, to form non-linear k-barrier coverage in mobile sensor networks. Simulation results show that the two proposed algorithms can efficiently reduce the movements of mobile sensors compared to a linear barrier algorithm and can be applied to large scale sensor networks. In the future, we will design a distributed algorithm for achieving non-linear k-barrier coverage using the horizontal virtual force.

**Acknowledgments.** This work is supported by National Natural Science Foundation of China (Grant No. 61702198). The corresponding author is Shuangjuan Li.

## References

1. Baheti, A., Gupta, A.: Non-linear barrier coverage using mobile wireless sensors. In: 2017 IEEE Symposium on Computers and Communications (ISCC). pp. 804–809. IEEE (2017)
2. Ban, D., Jiang, J., Yang, W., Dou, W., Yi, H.: Strong k-barrier coverage with mobile sensors. In: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference. pp. 68–72. ACM (2010)
3. Cheng, C.F., Wu, T.Y., Liao, H.C.: A density-barrier construction algorithm with minimum total movement in mobile wsns. Computer Networks 62, 208–220 (2014)
4. Cheng, T.M., Savkin, A.V.: A problem of decentralized self-deployment for mobile sensor networks: Barrier coverage between landmarks. In: 2009 IEEE International Conference on Control and Automation. pp. 1438–1442. IEEE (2009)
5. Du, J., Wang, K., Liu, H., Guo, D.: Maximizing the lifetime of $k$-discrete barrier coverage using mobile sensors. IEEE Sensors Journal 13(12), 4690–4701 (2013)
6. He, S., Chen, J., Li, X., Shen, X.S., Sun, Y.: Mobility and intruder prior information improving the barrier coverage of sparse sensor networks. IEEE transactions on mobile computing 13(6), 1268–1282 (2013)
7. Kim, H., Ben-Othman, J.: Heterbar: Construction of heterogeneous reinforced barrier in wireless sensor networks. IEEE Communications Letters 21(8), 1859–1862 (2017)
8. Kong, L., Liu, X., Li, Z., Wu, M.Y.: Automatic barrier coverage formation with mobile sensor networks. In: 2010 IEEE International Conference on Communications. pp. 1–5. IEEE (2010)
9. Kumar, S., Lai, T.H., Arora, A.: Barrier coverage with wireless sensors. In: Proceedings of the 11th annual international conference on Mobile computing and networking. pp. 284–298. ACM (2005)

10. Li, S., Shen, H.: Minimizing the maximum sensor movement for barrier coverage in the plane. In: 2015 IEEE Conference on Computer Communications (INFOCOM). pp. 244–252. IEEE (2015)
11. Qiu, C., Shen, H., Chen, K.: An energy-efficient and distributed cooperation mechanism for¡ inline-formula¿¡ tex-math notation. IEEE Transactions on Mobile Computing (6), 1247–1259 (2018)
12. Rout, M., Roy, R.: Self-deployment of randomly scattered mobile sensors to achieve barrier coverage. IEEE Sensors Journal 16(18), 6819–6820 (2016)
13. Saipulla, A., Liu, B., Xing, G., Fu, X., Wang, J.: Barrier coverage with sensors of limited mobility. In: Proceedings of the eleventh ACM international symposium on Mobile ad hoc networking and computing. pp. 201–210. ACM (2010)
14. Saipulla, A., Westphal, C., Liu, B., Wang, J.: Barrier coverage with line-based deployed mobile sensors. Ad Hoc Networks 11(4), 1381–1391 (2013)
15. Silvestri, S., Goss, K.: Mobibar: An autonomous deployment algorithm for barrier coverage with mobile sensors. Ad Hoc Networks 54, 111–129 (2017)
16. Tian, J., Liang, X., Wang, G.: Deployment and reallocation in mobile survivability-heterogeneous wireless sensor networks for barrier coverage. ad hoc networks 36, 321–331 (2016)
17. Wang, Z., Cao, Q., Qi, H., Chen, H., Wang, Q.: Cost-effective barrier coverage formation in heterogeneous wireless sensor networks. Ad Hoc Networks 64, 65–79 (2017)
18. Wu, Y., Cardei, M.: Distributed algorithms for barrier coverage via sensor rotation in wireless sensor networks. Journal of Combinatorial Optimization 36(1), 230–251 (2018)
19. Zhang, X., Wymore, M.L., Qiao, D.: Cost-efficient barrier coverage with a hybrid sensor network under practical constraints. In: 2017 IEEE International Conference on Communications (ICC). pp. 1–7. IEEE (2017)

**Zijing Ma** received a Bachelor's degree from South China Agricultural University. He will be a student at Central South University and study for a Master's degree in 2020. His main research interest is wireless sensor networks.

**Shuangjuan Li** is currently an teacher at South China Agricultural University, China. She received her BS and MS degrees in Wuhan University and her PhD degree in Sun Yat-sen University, China, in 2005, 2007, and 2016, respectively. Her current research interests include efficient algorithm design for wireless sensor network.

**Longkun Guo** received the B.S. and Ph.D. degrees in Computer Science from University of Science and Technology of China (USTC) in 2005 and 2011, respectively. He was a research associate of the University of Adelaide, and is currently a professor in School of Computer Science, Qilu University of Technology. His major research interest includes efficient algorithm design and computational complexity analysis for optimization problems in high performance computing systems and networks, VLSI etc. He has published more than 60 academic papers in reputable journals/conferences such as Algorithmica, IEEE Transactions on Mobile Computing, IJCAI, SPAA, etc.

**Guohua Wang** is currently an teaching fellow at South China Agricultural University, China. He received his BS degree in software engineering from South China Normal University and his MS and PhD degrees in computer software and theory from South

China University of Technology, China, in 2011, 2012, and 2016, respectively. His current research interests include computer vision, and pattern recognition.

*Received: February 5, 2020; Accepted: July 22, 2020.*

Computer Science and Information Systems 17(3):759–777 https://doi.org/10.2298/CSIS200216019C

# Maximize Concurrent Data Flows in Multi-radio Multi-channel Wireless Mesh Networks

Zhanmao Cao*,[1], Qisong Huang[1], and Chase Q. Wu[2]

[1] Department of Computer Science, South China Normal University
Guangzhou, Guangdong 510631, China
{caozhanmao,2018022617}@m.scnu.edu.cn
[2] Department of Computer Science, New Jersey Institute of Technology
Newark, New Jersey 07102, USA
chase.wu@njit.edu

**Abstract.** Multi-radio multi-channel (MRMC) wireless mesh networks (WMNs) have emerged as the broadband networks to provide access to the Internet for ubiquitous computing with the support for a large number of data flows. Many applications in WMNs can be abstracted as a multi-flow coexistence problem to carry out multiple concurrent data transfers. More specifically, links in different channel layers must be concatenated to compose multiple data transfer paths based on nodes' free interfaces and available channels. This is typically formulated as a combinatorial optimization problem with various stages including channel assignment, path computing, and link scheduling. This paper analyzes traffic behaviors and designs a coexisting algorithm to maximize the number of concurrent data flows. Simulations are conducted in combinatorial cases of channel and radio with various traffic requests of multiple pairs. The experimental results show the efficacy of the coexisting algorithm over a randomly generated topology. This scheme can be used to develop routing and scheduling solutions for various multi-flow network applications through prior computing.

**Keywords:** Concurrent flows, routing and scheduling, coexisting links, Wireless Mesh Networks.

## 1. Introduction

With rapid development of smart mobile terminals, data traffic requests converge to mesh routers, which form multiple data flows between multiple pairs. Wireless Mesh Networks (WMNs) have inherent advantages to serve ubiquitous communication as a broadband backbone. Increasingly, Multi-Radio Multi-Channel (MRMC) WMNs are believed to be the next-generation wireless backbone to address the challenge of heavy data flows [18]. Almost all applications in such network environments can be modeled as multi-flow tasks in a global wireless mesh topology. The fast growing demand for throughput in MRMC WMNs has led to a fundamental problem: how to assemble links to support multiple data flows with optimal performance in a coexisting manner, taking into consideration the limited resources in the network?

For traffic requests of multiple pairs $(s_i, d_i), i = 1, 2, ..., n$, a basic requirement is to enable continuous transmission without wireless interference. This is a fundamental and challenging problem that naturally boils down to a coexistence problem of concurrent flows, which must coexist and be activated along the paths for these specific pairs. Hence, the problem has a general meaning for different applications. As the data volume continues to increase, many applications need to transmit data between different pairs of nodes and it has become a common requirement to support concurrent flows. Due to the limited transmitting media and the wireless interference that obstructs the use of radios, it is challenging to design an efficient method that considers various factors simultaneously, such as interfaces, channels, interference, topology, traffic requests, and data size. To find the link group for optimal throughput in given deployment is meaningful, which help to reach a goal of capacity augmentation [3].

The challenge also arises from the problem's computational complexity. If we consider maximizing the number of concurrent flows or the throughput of source-destination pairs, the problem has been proved to be NP-complete [23]. A subproblem to decide a link schedule for optimal utilization of wireless resources has also been proved to be NP-hard [24]. Even in a simpler combinatorial case of multi-radio networks to meet a given set of rate demands through congestion control by considering channel assignment (CA) and traffic allocation, the problem still remains NP-hard [10]. Energy efficiency is another major concern as massive flows consume significant energy [26].

Cao *et al.* designed a joint routing and scheduling algorithm for multi-pair traffic requests based on a Cartesian Product of Graphs (CPG) model [5]. For the multi-pair or multi-flow problem, Cao *et al.* explored and addressed various issues in resource-aware situation [6]. They also designed a combinatorial scheme with consistent coexisting links over a topology of 64 nodes [11], and a co-existing scheme with optimal joint routing and scheduling [4]. More research efforts are needed in this line of research to support concurrent data flows in WMNs with limited resources.

Since the shortest-distance path does not account for the MRMC constraint, those well-known schemes such as Ad hoc On-Demand Distance Vector (AODV) routing cannot be directly applied to maximize the utilization of MRMC resources. To develop a concurrent transmission scheme for concurrent flows, we focus on the most important factors, including the radios $R$ of nodes, the set $C$ of available orthogonal channels, time slot $t$, topology $G$, and traffic requests $T_r$. This research sheds lights on several key points as follows:

- Formulate the multi-flow problem as a combinatorial optimization problem to achieve the maximum capacity of coexisting links over different channel layers under a channel decomposition model of CPG.
- Develop a link coexisting algorithm to support concurrent flows simultaneously.
- Evaluate the performance in terms of activated links, throughput, capacity, and delay in combinatorial situations.

The remaining sections are organized as follows: Section 2 provides a brief summary of related work. Section 3 builds an optimization model for maximum capacity and designs a coexisting algorithm. Section 4 evaluates the performance of the coexisting algorithm. Section 5 draws a conclusion of the work.

## 2. Related Work

Channel Assignment (CA) has been investigated in-depth through the use of graph coloring for interference avoidance. Almasaeid *et al.* proposed receiver based CA [2], in which they adjust the power and CA scheme in a strong restriction of cognitive radio WMN. The impact of network topology on channel resource utilization has been well recognized [22]. Interference-aware topology control has also been extensively studied for years [13,21]. Resource utilization also depends on the communication scheme. MRMC is able to significantly improve network capacity and reduce the cost of broadband WMN deployment [19]. The number of orthogonal channels and the number of radios per node determine the mesh capacity. The combined cases of critical resources should be compared to understand the relationship between throughput and other metrics.

Most of the existing work considered one aspect of routing, scheduling, or channel assignment. There is some limited work considering two or more of them. Jin showed that the routing and packet scheduling problem in general graphs is NP-complete [14]. It is even more challenging to optimize the use of wireless resources in WMNs, because one subproblem to determine an optimal link schedule has been shown to be NP-hard [24]. The subproblem only considering CA in mesh networks is similar to the least coloring problem of graph, which is NP-complete [23]. The joint routing and scheduling problem in WMNs is obviously more complicated. Even in a directional radio case, the transmission of multiple concurrent flows, which can be formulated as a mixed integer nonlinear problem, is inherently difficult to solve [25].

Alicherry *et al.* formulated CA and routing as an *LP* problem by simultaneously considering the characteristics of interference, the number of channels and the number of radios [1]. Giannoulis *et al.* proposed an iterative method to optimize congestion control by considering CA and traffic allocation [10], and declared that the problem is still NP-hard, even in a simpler combinatorial case on CA for multi-radio networks to realize a given set of rate demands. After decomposing congestion control into two stages, they formulated a problem of MRMC congestion control (MRMC-CC). However, Giannoulis' optimization method does not account for CA concerning multi-pair requests in MRMC WMN scenarios. Taking one step further from the above work, we investigate the problem of joint routing, scheduling, and CA with resource-awareness under the guidance of a carefully designed network model.

Due to limited resources, the shortest-path routing is insufficient for WMNs [8]. To find the critical links by using Bayesian theorem, which are bottleneck links in a WMN, was reported with significant performance improvement [12]. JRCA-AODV is reported as a joint routing and channel scheme, which is a modification of AODV. The problem of counting all shortest paths in an MIMO triangular mesh, considering the number of interfaces and the number of channels, is studied for WMNs [7]. The shortest-path routing scheme only considers the least resource consumption for one stream, while neglecting the fact that overlapped nodes may exhaust resources quickly [15]. If one node has no free resource, it cannot forward any packet. Kim *et al.* discussed resource sharing by quantifying node resource usage [16]. Although there exist some efforts in this direction, resource-aware routing still remains largely unexplored.

To simplify CA, Cao *et al.* suggested a virtual model with Cartesian product of graphs (CPG), which decomposes the complex layered structure [7]. Cao *et al.* developed a Cartesian product of graphs (CPG) model to simplify channel assignment. They proposed a destination-oriented routing method over a triangular mesh. Furthermore, they counted the path number with CPG model. There is very limited study on the efficiency of combinatorial cases considering the number of radios, the number of channels, network topology, time slots, and traffic distributions simultaneously. Cao *et al.* addressed combinatorial routing using CPG model, which is conducive to CA. The goal of CA is take full usage of the radio resource, which is represented in interference-free or more coexisting active links at a moment. Furthermore, they proposed optimal schemes of combinatorial routing and scheduling for concurrent flows in their recent work [4]. The considered factors in the model in the latest work [11], i.e., the major parameters, are of available channels, radios equipped, topology, and traffic requests.

To avoid interferences between links and to reduce heavy congestion on intersected nodes, path finding/selection and link scheduling should be carried out based on actual available resources. Hence, a joint scheduling and routing scheme with CA is deeply coupled with the network topology, the number of node interfaces, and the number of available channels.

## 3. Coexisting patterns

A data flow is carried over a path between a node pair $(s_i, d_i)$, where $s_i$ denotes the source node, and $d_i$ denotes the destination node. A traffic request from one mesh node $s_i$ to another $d_i$ with data size $z_i$, $i = 1, 2, ..., \rho$. Traffic mode is denoted as $T_r = \{(s_i, d_i; z_i) | i = 1, 2, ..., \rho\}$. The set of all traffic requests within a given period defines the traffic situation $T_r$. Typical examples include FTP or some other real-time data transfer requests.

We can depict this data transfer process in a general way with the help of the Cartesian Product of Graphs (CPG) model. Let each channel layer select a sufficient number of links for composing data transfer paths, which are candidates for routing the traffic of pair $(s_i, d_i)$. Note that, as the number of data flows increases, the problem becomes very challenging due to the limited resources. To support the transmission for multi-pair traffic requests, we need to analyze several main factors. According to the CPG model, a routing and scheduling scheme should consider the number $|R|$ of radios, the number $|C|$ of available channels, the topology $G$ of the multi-radio multi-channel (MRMC) wireless mesh network (WMN), and traffic mode $T_r$. The four major factors are deemed to have main impacts on the WMN performance. The first two factors are main resources, as time is naturally considered when scheduling selected links. Topology is the base for identifying the interference relationship. In wireless mesh networks, the network topology should be considered in the beginning of the network deployment. Considering four factors simultaneously is far more complex compared with most of the existing work that considers only one or two factors, as in cognitive networks [20].

To take full advantage of MRMC resources, we need to focus on a certain performance metric for the mesh. In this work, we consider the maximum capacity for traffic requests in a given WMN as the optimization goal under certain constraints on the paths for routing or the links to be scheduled. We first explore the characteristics of links, combinatorial conditions, and active link numbers, and then formulate a combinatorial optimization problem. We design an algorithm to create coexisting paths for this problem and evaluate the performance of the algorithm in various combinatorial scenarios.

### 3.1. Model for Coexisting Links

We consider a WMN topology $G = (V, E), |V| = n, |E| = m$, where $E$ denotes the set of effective communications between neighbor nodes, not actual links in the wireless mesh. Only when a pair of nodes $(u, v)$ in $V$ are communicating with each other over the same channel $c_j$, $(u, v) \in E$ becomes a link at an assigned time slot $t$, denoted as $l_{c_j,t}^{(u,v)}$. If the channel $c_j$ has bandwidth $\omega_j$, the maximum capacity of the link is $\omega_j$. Obviously, the flow data rate of $l_{c_j,t}^{(u,v)}$ is bounded by $\omega_j$.

The maximum capacity depends on several critical factors: the traffic situation $T_r$, the resources of the mesh represented by $R$ and $C$, the topology, and the scheme for routing and scheduling concurrent flows of different pairs. $T_r = \{(s_i, d_i; z_i)|i = 1, 2, ..., \rho\}$ is the initial traffic situation. The resources of a mesh node $v_i$ include radio number $R[i]$ of and the set $C = \{c_1, c_2, ..., c_q\}$ of available channels, where $q$ is the total number of orthogonal channels. The topology is another important input of the problem as it affects route selection and node/link interference relation.

The path of node pair $(s_i, d_i)$ is denoted as $P_{(s_i,d_i)}$, or simply as $P_i$. The number of hops along $P_i$ from $s_i$ to $d_i$ is denoted as $\varrho_i$. We denote the $j^{th}$ hop of $P_i$ as $\hbar_j^{P_{(s_i,d_i)}}$, or $\hbar_j^{P_i}$ for brevity. The channel assigned to link $\hbar_j^{P_i}$ is denoted as $c_{ij} \in C$, and its corresponding bandwidth is denoted as $\omega_{ij}$. The capacity of $P_{(s_i,d_i)}$ is the sum of link capacities along $P_i$, denoted as $Cap(P_{(s_i,d_i)})$. The lower bound of $Cap(P_{(s_i,d_i)})$ is the minimum link capacity along $P_i$ multiplied by $\varrho_i$. Similarly, the upper bound of $Cap(P_{(s_i,d_i)})$ is the maximum link capacity multiplied by $\varrho_i$. The capacity of an active path $P_i$ is calculated as:

$$Cap(P_{(s_i,d_i)}) = \sum_{j=1}^{\varrho_i} \omega_{ij}. \tag{1}$$

Over a certain channel layer in the CPG model, the maximum number of links is determined by the topology. The choice of maximum links may not be unique, but the maximum number of coexisting links must match the number of node-pairs at all times. Let $\lambda$ be the maximum number of coexisting links in one channel layer. Then, the number of possible links over all $q$ channels in a given mesh can be estimated by its upper bound $q \cdot \lambda$. In fact, it is critical to cooperative arrange the channel for efficient scheduling as mentioned in paper [9].

However, the traffic situation may contain not only one-hop communications, but also many-hop communications for $P_i$. Generally, we need to concatenate several links, which are distributed in different channel layers, to compose the paths for the current $T_r = \{(s_i, d_i; z_i)|i = 1, 2, ..., \rho\}$. We have $\hbar_{j,c_k,t}^{P_i} = 1$, if the link is scheduled; else, $\hbar_{j,c_k,t}^{P_i} = 0$. Obviously, the capacity is also limited by the size of $T_r$. Hence, the maximum capacity, as the optimization objective, is represented by

$$\max \sum_{i=1}^{\rho} \sum_{j=1}^{\varrho_i} \sum_{k=1}^{q} \sum_{t=1}^{T} \hbar_{j,c_k,t}^{P_i} \cdot \omega_k. \tag{2}$$

Clearly, the link count of a node is limited by its radios. If node $v_i$ is a link's receiver over $c_j$, we denote this link as $l_{c_j,t}^{\triangleright,v_i}$; if node $v_i$ is a link's sender, we denote this link as $l_{c_j,t}^{v_i,\triangleright}$. Hence, $l_{c_j,t}^{\triangleright,v_i} + l_{c_j,t}^{v_i,\triangleright} \leq R[i], \forall i$. Here, $\triangleright$ denotes the direction of the radio from/to a certain node.

If there are several paths sharing one connection over channel $c_j$, say $l_{c_j,t}^{(u,v)}$, then the minimum sum of the link capacities for those sharing paths must be less than $\omega_j$.

In the CPG model, the links of a path are distributed in various channel layers as shown in [7]. The links in one channel layer are collected in a greedy way to have sufficient interference-free links for an initial path and other interference-free links to support other paths. The critical step is to find the best fit for path edges and interference-free links. However, the combinations of links in different layers are restricted by the paths of different pairs. The combinatorial nature makes the problem extremely challenging.

We attempt to design a heuristic scheme to find as many required links as possible and generate multiple coexisting link groups. Based on the model and the coexisting properties discussed above, we design our scheme as shown in Algorithm 1, where the coexisting links are combined to maximize the number of coexisting paths.

### 3.2. The Link Coexistence Algorithm

To simultaneously activate as many paths as possible, it is necessary to make full use of the network resources. In the CPG model, we understand that one-hop paths cannot always meet the demands for concurrent flows in practice. Let $C(v_i)$ denote the set of all available channels of router $v_i$, and let $c_j(v_i)$ denote the channel assignment operation, which assigns channel $c_j$ to router $v_i$. We use $I_{Free}$ to denote the set of links without interference over a certain channel, i.e., every two links in $I_{Free}$ satisfy the interference-free relation. In fact, for a specific topology, $I_{Free}$ holds over any channel.

We use $L_t$ to denote the set of all links that satisfy $I_{Free}$ at time $t$, i.e., $L_t \bowtie I_{Free}$. Here, $\bowtie$ means "satisfy the right-side relation". If new links are added into $L_t$, the interferences have to be examined.

Based on the models constructed in Section 3.1, we design Algorithm 1 to optimize resource utilization. It produces the maximum link groups at time $t$. As a result, the links in $L_t$ are combined to form as many paths as possible for multiple pairs $(s_i, d_i), i = 1, 2, ..., \rho$.

## 4. Performance Evaluation

Topology is the first factor to be considered in wireless mesh networks, because it determines the interference relation between routers. To evaluate the performance of Algorithm 1, we conduct a set of simulations in combinatorial cases $R \times C = \{4, 8, 12, 16, 20\} \times \{8, 16, 32, 64\}$ with an MRMC WMN topology, as shown in Fig. 1. This topology has 77 nodes, which are generated randomly in NS3 with distance constraints.

### 4.1. Simulations in Combinatorial Cases

The performance of a WMN depends on several critical factors: the traffic situation $T_r$, the resources of the mesh represented by $R$ and $C$, the topology, and the scheme for routing and scheduling concurrent flows of different pairs. Given a region, the router deployment is typically fixed. Hence, we consider one topology in our simulations. In addition, we do not include any base station (BS) in the network topology. Traffic $T_r$ is selected from four groups of different traffic types, each of which has 80 pairs. Each $T_r$

**Algorithm 1** Link coexistence algorithm for concurrent flows

Input: Topology $G$, the set $R$ of node radios, and the set $C$ of available channels, traffic mode $T_r = \{(s_i, d_i; z_i) | i = 1, 2, ..., \rho\}$.

Output: $L$, the set of coexisting links combined to support concurrent flows.

```
Require: $|R| \geq 0 \wedge |C| \geq 0$;
Ensure: $\omega_i \geq 0$;
$t := 0$;
$i := 1$;
$j := 1$;
for ($t = 0$ to $T$) do
    while ($i < \rho$) do
        if ($\exists$ path $P_i \wedge \forall v_h \in P_i, r(v_h) > 0$) then
            if ($\forall i, h, (v_h \in P_i) \wedge (v_h \neq s_i) \wedge (v_h \neq d_i) \wedge (c(v_h) > 0)$) then
                Choose $c_j \in C(v_{i_{h-1}}) \cap C(v_{i_h})$, and let $c_j(l_t^{(v_{i_{h-1}}, v_{i_h})})$;
                if $L_t \cup_i \{l_{c_j,t}^{(v_{i_{h-1}}, v_{i_h})}\} \bowtie I_{Free}$ then
                    $L_t := L_t \cup \{l_{c_j,t}^{(v_{i_{h-1}}, v_{i_h})}\}$;
            else $\{P_i$ has a node with no free resources$\}$
                $i := i + 1$;
    $t := t + 1$;
output $L := \bigsqcup_t \{L_t\}, \{t \in (0, T)\}$.
```

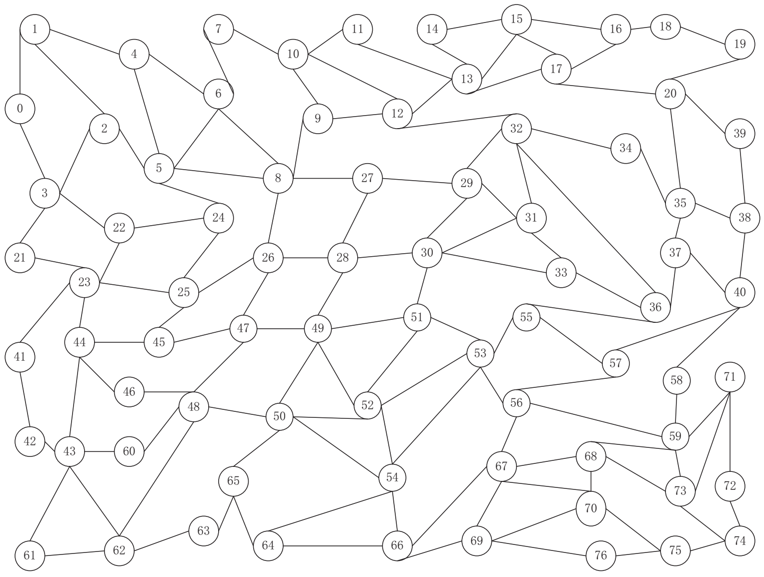


**Fig. 1.** The WMS topology with 77 routers.

is randomly generated between $(s_i, d_i)$ pair with data size $z_i > 0$. $R$ and $C$ are critical resources in WMNs, playing a significant role in the stage of path planning and routing. The combinatorial cases are constructed based on $R \times C$.

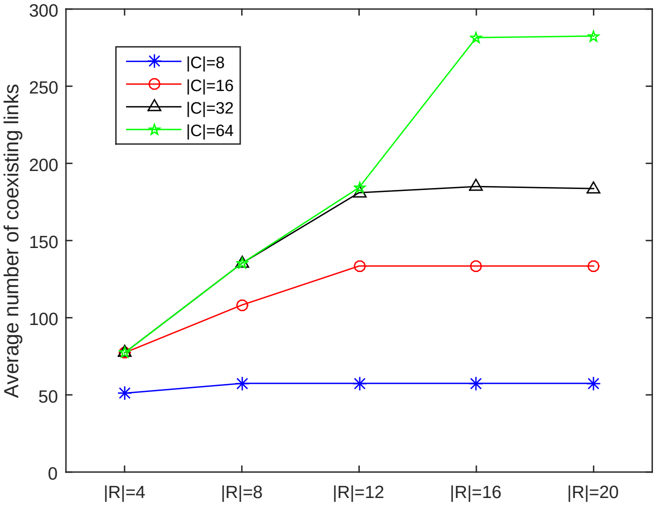


**Fig. 2.** The average number of activated links in the topology.

In the topology as shown in Fig. 1, we have $C = \{8, 16, 32, 64\}$, $R = \{4, 8, 12, 16, 20\}$, and $|T_r| = 80$. We first need to find out the number of simultaneously activated links. The number of coexisting links for the combinations is plotted in Fig. 2. As links for different pairs are simultaneously activated, the more links are activated, the more packets are forwarded. The simulation also presents an interesting phenomenon: when $|R| = 12$, channel count $|C| = 32$ is sufficient to use the interface resources. Having more channels may not improve the number of simultaneous links with $|R| = 12$. Also, increasing the number of interfaces may not improve the number of simultaneous links with $|C| = 32$, as shown in Fig. 2. We refer to the situation $|R| = 12 \wedge |C| = 32$ as a hand-shake match. Similarly, $|R| = 16 \wedge |C| = 64$ is another hand-shake match. The best use of free resources may sustain nearly up to 290 links in the given mesh.

The average capacities are measured to evaluate the performance of the coexisting algorithm for the combinations of $R \times C = \{4, 8, 12, 16, 20\} \times \{8, 16, 32, 64\}$. The multi-pair requests form a random group of 80 pairs, i.e., $|Tr| = 80$. Generally, more resources equipped in the mesh promise a higher capacity. However, we note that if there are only 8 available channels, the capacity cannot be improved further by increasing the number of radios. Furthermore, in the case of $|C| = 16$ or $|C| = 32$, the upper capacity limit can be reached with $|R| = 12$. Similarly, in the case of $|C| = 64$, the upper capacity limit can be reached with $|R| = 16$. Generally, the mesh provides a large capacity as shown in Fig. 3.

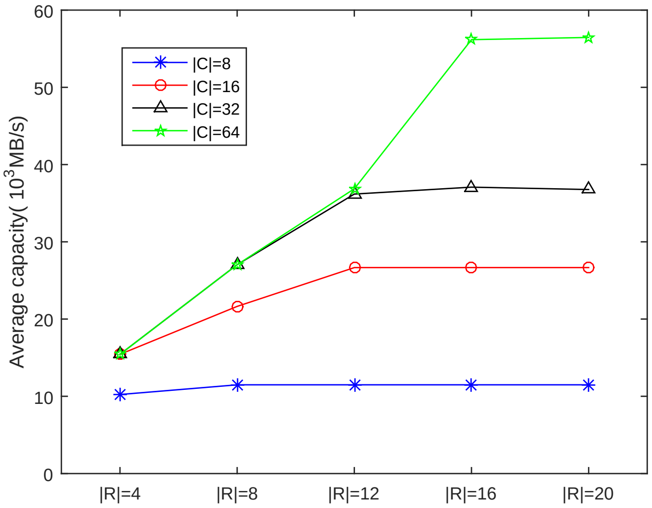


**Fig. 3.** The average capacity with $R \times C = \{4, 8, 12, 16, 20\} \times \{8, 16, 32, 64\}$.

The above two simulations are conducted with traffic requests $|Tr| = 80$, while the data size is set to be sufficiently large for continuous transmission during one period. The topology is randomly generated with 77 nodes under distance constraints. The simulations are based on the combinations of resources $R = \{4, 8, 12, 16, 20\}$ and $C = \{8, 16, 32, 64\}$.

The following simulation is designed to examine the performance of the proposed coexisting algorithm. Note again that there are no base stations in the topology. Further simulations are conducted to measure the performance in terms of average throughput, average activated link number, and average capacity of the whole network. We consider different combinatorial cases of $C \times R = \{8, 16, 32, 64, 128\} \times \{4, 8, 12, 16\}$. Here, we reduce the scenario of radio number, while increasing that of channels. Resource utilization is estimated via link number, capacity, throughput in $R \times C$ combinatorial cases, with $|T_r| = 80$ of four types of traffic requests.

As described in the model analysis, the algorithm finds coexisting links and combines them to form paths of different pairs. The links are activated simultaneously in a time slot without interference. This suggests counting the average number of activated links during a time period for performance evaluation. The more links are activated, the better the performance it should be. The performance measurements are plotted in Fig. 5.

We observe that the number of activated links increases with the resources. However, for a certain $|R|$, the average link number has an upper bound even if $|C|$ is doubled. This can be verified in cases of $|R| = 4$ with $|C| = 16$, $|R| = 8$ with $|C| = 32$, and $|R| = 12$ with $|C| = 64$. In particular, $|R| = 16$ with $|C| = 64$, where $|C|$ is doubled to 128. Here, the activated link number does not increase accordingly. This observation indicates that radios and channels may have some match pattern in the performance. To deploy a local-

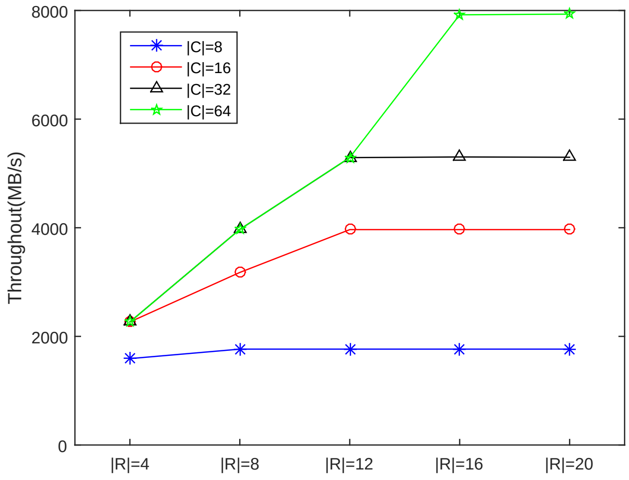


**Fig. 4.** The average throughput with $C \times R = \{8, 16, 32, 64\} \times \{4, 8, 12, 16, 20\}$.

area WMN, a careful pre-computation to find the matching pattern may save engineering time and investment cost.

A combinatorial simulation is shown in Fig. 6. Note that there are some patterns between radio number and channel number, as the throughput reaches a plateau at a certain channel number for one radio number, because the radio number has a certain limitation to the performance. For example, when $|R| = 4$, the performance has a jump from $|C| = 8$ to $|C| = 16$. But there is no improvement even if the channel number is increased to 128. We refer to this situation as a match of $R$ with $C$ from the view of performance. A similar pattern repeats for $|R| = 8, 12$, and 16. Correspondingly, $|R| = 8, 12$, and 16 match with $|C| = 32, 32$, and 64, respectively. This calls for further simulations or comparisons in those matching cases.

In one scheduling period, as the combinatorial routing chooses different link patterns according to different traffic requests, the actual capacity may change with the heuristic start and traffic requests. We further conduct experiments to evaluate the robustness of the algorithm in terms of the average capacity. The simulations are conducted with four groups of 80 pair traffic requests each over the topology in Fig. 1, in the combinatorial resource cases of $C \times R = \{8, 16, 32, 64, 128\} \times \{4, 8, 12, 16\}$. This may also help determine if the simulations for throughput and capacity are consistent with each other. If they are consistent, the results would support the robustness of the proposed algorithm. The performance measurements are plotted in Fig. 7, which shows that the results do match the average throughput.

The simulations show that the number of interfaces $R$ and the number of available channels have a certain fitness. In the case of the average activated links with $C \times R = \{8, 16, 32, 64, , 128\} \times \{4, 8, 12, 16\}$, as shown in Fig.4, we understand that the efficiency of channel resource is limited by the number of node interfaces. In the case of $|R| = 12$, the performances are the same for $|C| = 32$ and $|C| = 64$. This phenomenon indicates

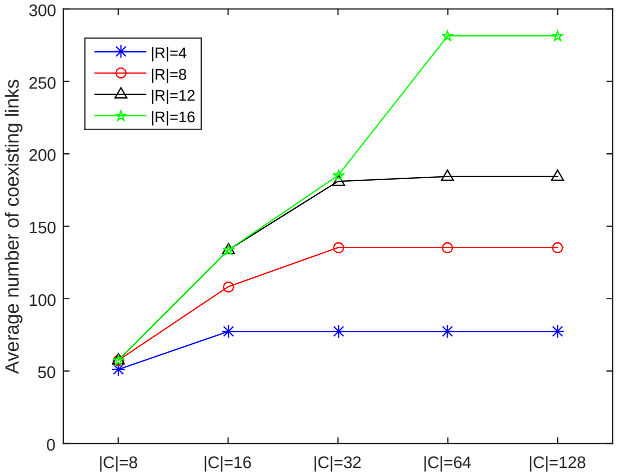


**Fig. 5.** The average number of activated links with $C \times R = \{8, 16..., 128\} \times \{4, ..., 16\}$.

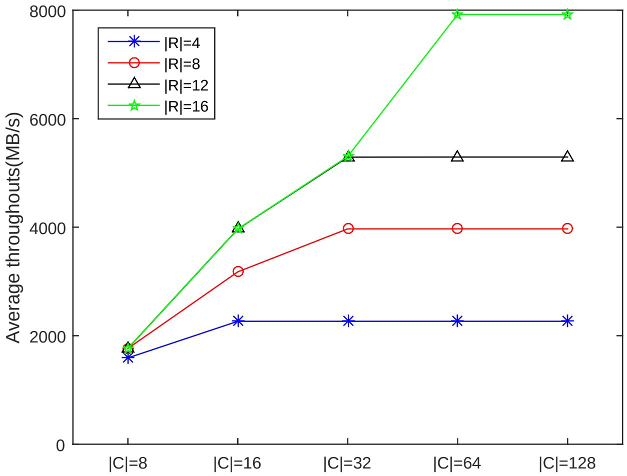


**Fig. 6.** Average throughput with $C \times R = \{8, 16, 32, 64, 128\} \times \{4, 8, 12, 16\}$.

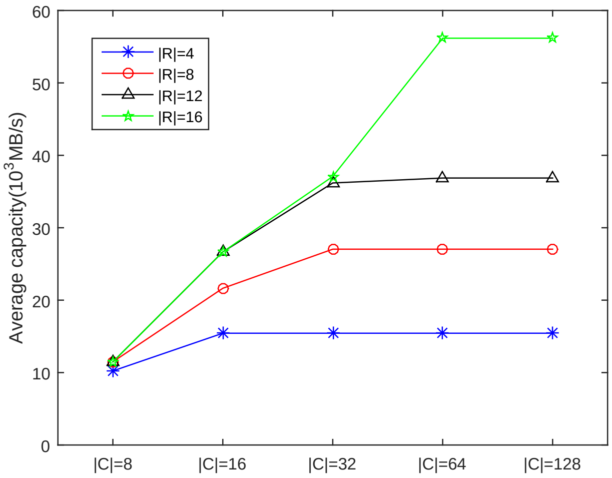


**Fig. 7.** Average capacity with $C \times R = \{8, 16, 32, 64, 128\} \times \{4, 8, 12, 16\}$.

that the most efficient number of channels is 32, as the throughput is upper bounded at $|C| = 32$. In this topology, $|R| = 12$ may be combined with $|C| = 32$ as the most efficient pattern. Allocating more channels to the given deployment region may result in the waste of resource.

The simulation results show that there exists a best match of $|R| = 16\&|C| = 64$. At this match, we achieve almost the same highest throughput, for both combinatorial cases $R \times C = \{4, 8, 12, 16, 20\} \times \{8, 16, 32, 64\}$ and $C \times R = \{8, 16, 32, 64, 128\} \times \{4, 8, 12, 16\}$. We also observe that the experimental results are consistent in both cases of different resource deployment schemes, which illustrates the robustness of Algorithm 1.

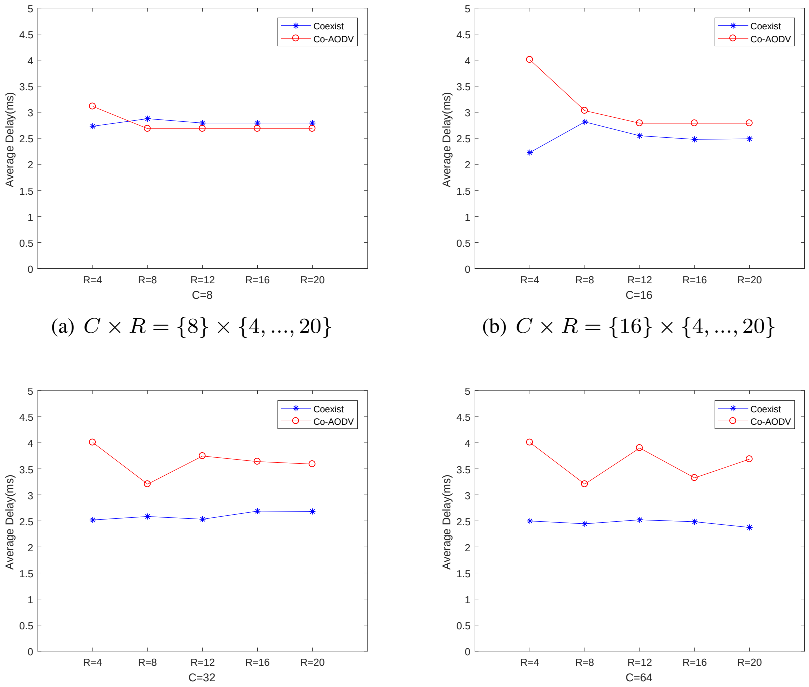
### 4.2. Comparison Experiments

We compare the proposed scheme with AODV for performance evaluation. As AODV can not be directly applied to the MRMC situation, we modify AODV to work in the multiple-channel situation, which is referred to as co-AODV. The comparisons are divided into three groups, which are shown in three figures with subgraphs. Fig. 8, Fig. 9, and Fig. 10 plot the comparison results of delivery delay, packet delivery ratio, and throughput, respectively. In each group, the comparison is made in four subcases: 1) $C \times R = \{8\} \times \{4, 8, 12, 16, 20\}$, 2) $C \times R = \{16\} \times \{4, 8, 12, 16, 20\}$, 3) $C \times R = \{32\} \times \{4, 8, 12, 16, 20\}$, and 4) $C \times R = \{64\} \times \{4, 8, 12, 16, 20\}$.

In Fig. 9, we measure the packet delivery ratio ($PDR$), which is defined as the total number of successfully received packets $p_{rece}$ divided by the total number of sent packets $p_{send}$, i.e.,

$$PDR = \frac{p_{rece}}{p_{send}} \times 100\%. \tag{3}$$

The average PDR measurements show that Algorithm 1 works efficiently for MRMC WMNs. With less resources as shown in Fig. 9(a), it achieves a performance close to



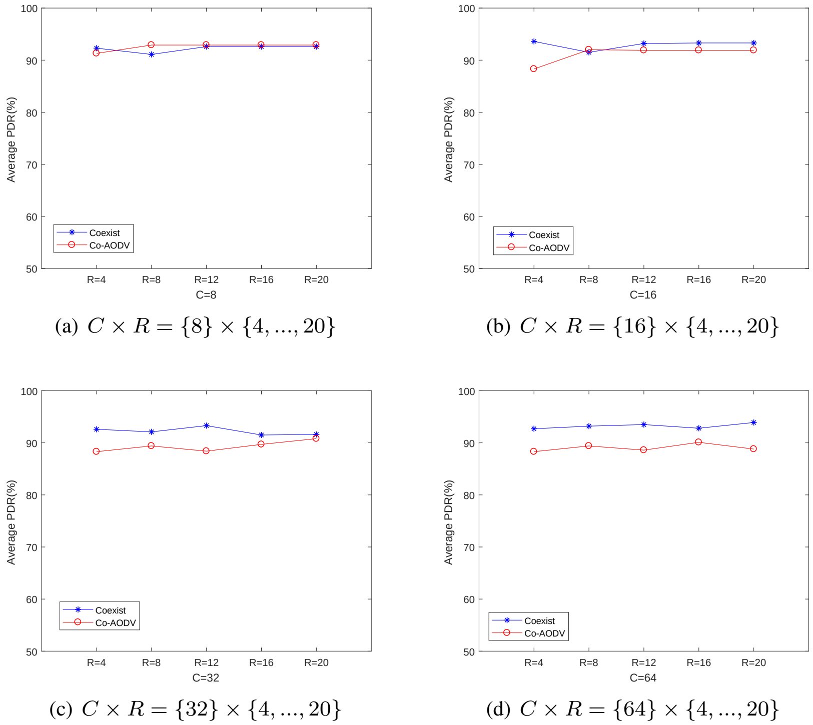
(a) $C \times R = \{8\} \times \{4, ..., 20\}$

(b) $C \times R = \{16\} \times \{4, ..., 20\}$

(c) $C \times R = \{32\} \times \{4, ..., 20\}$

(d) $C \times R = \{64\} \times \{4, ..., 20\}$

**Fig. 8.** Average delay comparison with AODV.



(a) $C \times R = \{8\} \times \{4, ..., 20\}$

(b) $C \times R = \{16\} \times \{4, ..., 20\}$

(c) $C \times R = \{32\} \times \{4, ..., 20\}$

(d) $C \times R = \{64\} \times \{4, ..., 20\}$

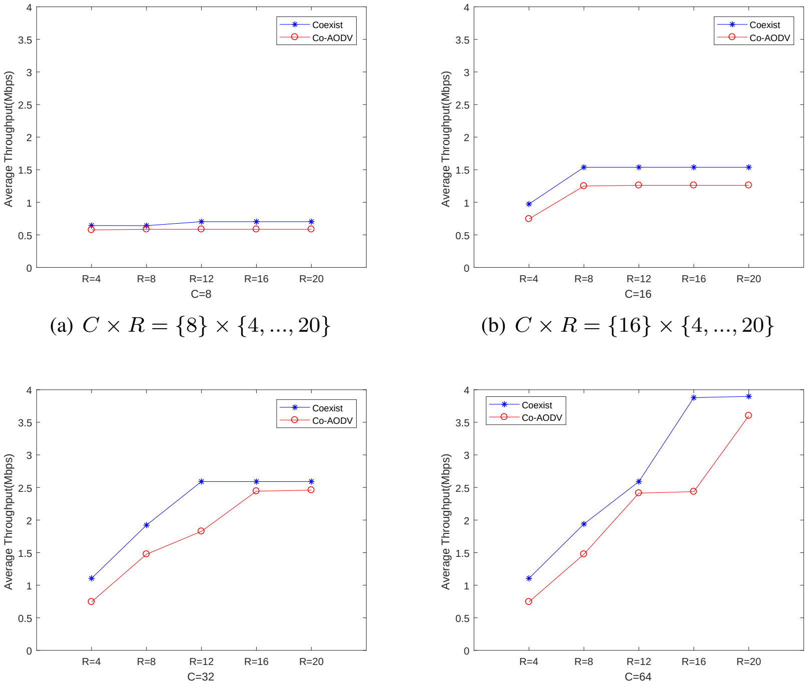
**Fig. 9.** Comparison of packet delivery ratio (PDR) with AODV.

AODV. With more resources, it increasingly outperforms AODV as shown in Figs. 9(b), 9(c), and 9(d).

We also compare our algorithm with AODV in terms of throughput. Similarly, we conduct experiments in four sub-problems as above, but with focus on throughput performance.

These results show that Algorithm 1 outperforms AODV in terms of average throughput in the combined cases, which illustrate the resource utilization efficiency of the proposed algorithm. We observe a $R \times C$ pattern where the throughput jumps at $C = 32 \; when \; R = 12$ as shown in Fig. 10(c) and $C = 64 \; when \; R = 16$ as shown in Fig. 10(d). This observation suggests a deployment scheme with matched radios and channels in this given region.

In order to investigate the performance of our proposed scheme, it is necessary to compare in maximum throughput, see Fig. 11. We conducted a group of combinatorial comparisons. The simulation results show that co-AODV works well when there are less resources. Meanwhile, the coexisting scheme outperforms significantly with $C = 32, C = 64$ and $R = 8, 12, 16, 20$. Notice that the simulations can also be affected by traffic situations, the efficiency assessment of an algorithm should primarily depends on the average throughput.



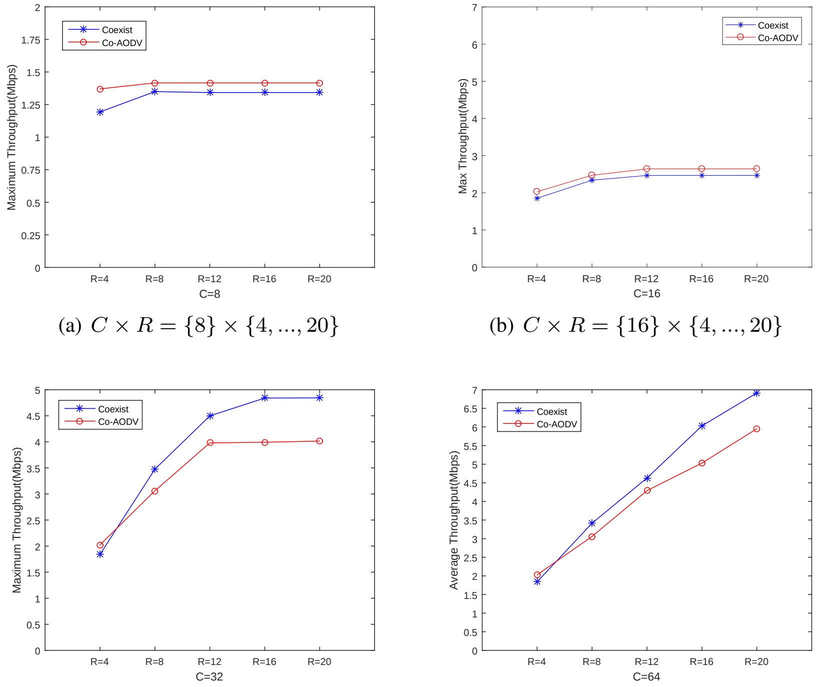
(a) $C \times R = \{8\} \times \{4, ..., 20\}$

(b) $C \times R = \{16\} \times \{4, ..., 20\}$

(c) $C \times R = \{32\} \times \{4, ..., 20\}$

(d) $C \times R = \{64\} \times \{4, ..., 20\}$

**Fig. 10.** Average throughput comparison with AODV.



(a) $C \times R = \{8\} \times \{4, ..., 20\}$

(b) $C \times R = \{16\} \times \{4, ..., 20\}$

(c) $C \times R = \{32\} \times \{4, ..., 20\}$

(d) $C \times R = \{64\} \times \{4, ..., 20\}$

**Fig. 11.** Maximum throughput comparison with co-AODV.

## 5. Conclusion

Based on the channel layered CPG model, we formulated the optimal capacity problem for concurrent flows in MRMC WMNs as a combinatorial optimization problem to maximize the number of coexisting links over different channel layers. We analyzed the optimization objective and the corresponding constraints, and designed a link coexisting algorithm to meet the demands for maximum transmission. The proposed algorithm is based on a heuristic greedy strategy, which specifically accounts for the complexity of concatenating links into paths.

To approach the optimal performance, combinatorial techniques are used to decompose coexisting paths into channel layers. Simulation results show that this algorithm provides an effective solution by making full use of available resources. Three sets of simulations illustrate the robustness of the algorithm and the performance improvement in comparison with AODV, in terms of delay, PDR, and average throughput. The network topology and $\{Tr, R, C\}$ together determine a joint routing and scheduling scheme, which can be used to pre-compute routes and schedules for various multi-flow tasks. Considering the complexity of this problem, as we noticed there are attempts by machine learning in the field, such as work [17], using machine learning in WMNs may be a new direction in our future research.

## References

1. Alicherry, M., Bhatia, R., Li, L.E.: Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks. In: Proceedings of the 11th Annual International Conference on Mobile Computing and Networking. pp. 58–72. ACM MobiCom '05 (2005)
2. Almasaeid, H.M., Kamal, A.E.: Receiver-based channel allocation in cognitive radio wireless mesh networks. IEEE/ACM Transactions on Networking 23(4), 1286–1299 (2015)
3. Ashraf, U.: Capacity augmentation in wireless mesh networks. IEEE Transactions on Mobile Computing 14(7), 1344–1354 (2015)
4. Cao, Z., Huang, Q., Wu, Q.C., Kong, W., Hou, A.: On a coexisting scheme for multiple flows in multi-radio multi-channel wireless mesh networks. In: Communications in Computer and Information Science 1163. pp. 3–11. Springer (2019)
5. Cao, Z., Wu, C.Q., Berry, M.L.: An optimization scheme for routing and scheduling of concurrent user requests in wireless mesh networks. Computer Science and Information Systems 14(3), 661–684 (2017)
6. Cao, Z., Wu, C.Q., Berry, M.L., Wang, Y.: Resource-aware routing and scheduling in multi-radio multi-channel wireless mesh networks. In: IEEE Wireless Communications and Networking Conference Workshops (WCNCW). pp. 248–253. International Workshop on Big Data with Computational Intelligence for Wireless Networking'2018 (2018)
7. Cao, Z., Wu, Q., Zhang, Y., Shiva, S.G., Gu, Y.: On modeling and analysis of mimo wireless mesh networks with triangular overlay topology. Mathematical Problems in Engineering p. Article ID 185262 (2015), http://www.hindawi.com/journals/mpe/aa/185262/
8. De Couto, D.S., Aguayo, D., Chambers, B.A., Morris, R.: Performance of multihop wireless networks: Shortest path is not enough. ACM SIGCOMM Computer Communication Review 33(1), 83–88 (2003)
9. Deng, X., Luo, J., He, L., Liu, Q., Li, X., Cai, L.: Cooperative channel allocation and scheduling in multi-interface wireless mesh networks. IEEE Transactions on Mobile Computing 12, 1–12 (2019)

10. Giannoulis, A., Salonidis, T., Knightly, E.: Congestion control and channel assignment in multi-radio wireless mesh networks. In: Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON'08. 5th Annual IEEE Communications Society Conference on. pp. 350–358. IEEE (2008)
11. Huang, Q., Cao, Z., Shan, Z.: Combinatorial routing and scheduling scheme for multiple concurrent flows in wireless mesh networks. Journal of South China Normal University (Natural Science Ed.) 52(1), 122–128 (2020)
12. Iqbal, S., Abdullah, A.H., Ahsan, F., Qureshi, K.N.: Critical link identification and prioritization using bayesian theorem for dynamic channel assignment in wireless mesh networks. Wireless Networks 24, 2685–2697 (2018)
13. Jang, H.C., Fang, R.W.: Interference-aware topology control in wireless mesh network. In: Proceedings of the International Conference on Mobile Technology, Applications, and Systems. pp. 1–7. MTAS'08 (2008)
14. Jin, F., Arora, A., Hwang, J., Choi, H.A.: Routing and packet scheduling in wimax mesh networks. In: Proc. of the 4th IEEE International Conference on Broadband Communications, Networks and Systems (BROADNETS). pp. 574–582 (2007)
15. Jun, J., Sichitiu, M.L.: Mrp: Wireless mesh networks routing protocol. Computer Communications 31(7), 1413–1435 (2008)
16. Kim, T.S., Yang, Y., Hou, J.C., Krishnamurthy, S.V.: Joint resource allocation and admission control in wireless mesh networks. In: Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 2009. WiOPT 2009. 7th International Symposium on. pp. 1–10. IEEE (2009)
17. Kuskonmaz, B., Ozkan, H., Gurbuz, O.: Machine learning based smart steering for wireless mesh networks. Ad Hoc Networks 88, 98–111 (2019)
18. Larsson, E.G., Edfors, O., Tufvesson, F., Marzetta, T.L.: Massive mimo for next generation wireless systems. IEEE Communications Magazine 52(2), 186–195 (2014)
19. Liu, Q., Jia, X., Zhou, Y.: Topology control for multi-channel multi-radio wireless mesh networks using directional antennas. Wireless Networks 17(1), 41–51 (2011)
20. Maheen, I., M. A., R., Md, Mamun-Or-Rashid, e.a.: Traffic engineering in cognitive mesh networks: Joint link-channel selection and power allocation. Computer Communications 116, 212–224 (2018)
21. Maleki, E.N., Mirjalily, G.: Fault-tolerant interference-aware topology control in multi-radio multi-channel wireless mesh networks. Computer Networks 110, 206–222 (2016)
22. Marina, M.K., Das, S.R., Subramanian, P.: A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks. Computer Networks 54, 241–256 (2010)
23. Ramachandran, K.N., Belding-Royer, E.M., Almeroth, K.C., Buddhikot, M.M.: Interference-aware channel assignment in multi-radio wireless mesh networks. In: INFOCOM. vol. 6, pp. 1–12 (2006)
24. Ramanathan, S., Lloyd, E.L.: Scheduling algorithms for multihop radio networks. IEEE/ACM Transactions on Networking (TON) 1(2), 166–177 (1993)
25. Roh, H.T., Lee, J.W.: Channel assignment, link scheduling, routing, and rate control for multi-channel wireless mesh networks with directional antennas. Journal of Communications and Networks 18(6), 884–891 (2016)
26. Shu, T., Wu, C.Q.: Energy-efficient mapping of large-scale workflows under deadline constraints in big data computing systems. Elsevier FGCS (2018), doi.org/10.1016/j.future.2017.07.050

**Zhanmao Cao** received his PhD in 2011, from School of Computer Science and Technology, South China University of Technology, Guangzhou, China. He is Associate Professor in School of Computer, South China Normal University. He is interested in developing algorithms for application problems, wireless mesh networks, bioinformatics, and machine learning. He proposed algorithms BTA and DC-BTA for multiple sequence alignment. He is now working on modeling, routing and scheduling algorithms for WMNs, as well as applications of machine learning.

**Qisong Huang** is a graduate student of grade 2018 in School of Computer, South China Normal University. His research is on WMN communication algorithms and simulations.

**Chase Wu** completed his Ph.D. dissertation at Oak Ridge National Laboratory and received his Ph.D. degree in computer science from Louisiana State University in 2003. He was a research fellow at Oak Ridge National Laboratory during 2003–2006 and an associate professor at University of Memphis during 2006–2015. He is currently a professor at New Jersey Institute of Technology. His research interests include big data, parallel and distributed computing, machine learning, high-performance networking, sensor networks, and cyber security.

*Received: February 16, 2020; Accepted: June 23, 2020.*

Computer Science and Information Systems 17(3):779–794 https://doi.org/10.2298/CSIS200204020Z

# An Improved MCB Localization Algorithm Based on Weighted RSSI and Motion Prediction

Chunyue Zhou[1], Hui Tian[2], and Baitong Zhong[3]

[1] Laboratory of Communication Engineering
Beijing Jiaotong University, China
zhouchunyue@hotmail.com
[2] School of Information and Communication Technology
Griffith University, Australia
hui.tian@griffith.edu.au
[3] Hunan Electronic Technology Vocational College
Hunan Province, China
471727138@qq.com

**Abstract.** Aiming at the problem of low sampling efficiency and high demand for anchor node density of traditional Monte Carlo Localization Boxed algorithm, an improved algorithm based on historical anchor node information and the received signal strength indicator (RSSI) ranging weight is proposed which can effectively constrain sampling area of the node to be located. Moreover, the RSSI ranging of the surrounding anchors and the neighbor nodes is used to provide references for the position sampling weights of the nodes to be located, an improved motion model is proposed to further restrict the sampling area in direction. The simulation results show that the improved Monte Carlo Localization Boxed (IMCB) algorithm effectively improves the accuracy and efficiency of localization.

**Keywords:** Wireless sensor networks, Localization, Monte Carlo Boxed, RSSI, Motion prediction.

## 1. Introduction

Wireless sensor networks (WSN) is a wireless communication network system that integrates monitoring, control, wireless communication and other functions. With the development of wireless sensor network technology, sensors are widely used in environment, military, medical, space exploration and many other fields. In these applications, the location of sensor nodes is a very important information, locating technology has become the key support technology of wireless sensor networks. With the diversification of wireless sensor network applications and the gradual maturity of static network node locating technology, mobile node locating technology has attracted more and more attention in recent years.

The localization algorithm of wireless sensor network nodes can be divided into localization algorithm based on ranging and localization algorithm based on non-ranging. Ranging-based localization algorithm is based on AOA [1], TOA [2], TDOA [3], RSSI [4] and other ranging technologies to obtain the distance between ordinary nodes and anchor nodes, and then use spatial geometry rules to calculate the location of unknown nodes through triangular localization algorithm, trilateral localization algorithm, multilateral localization algorithm, maximum likelihood estimation algorithm, etc. Non-ranging localization algorithm does not need to measure the distance between nodes, but calculates the location of unknown nodes by calculating the connectivity between unknown nodes and surrounding nodes, such as centroid localization [5], DV-Hop localization [6], convex programming [7], APIT [8], etc.

Fixed wireless sensor network node localization algorithms are numerous and relatively mature, but the complexity of mobile node localization algorithm leads to a huge amount of computation, and the localization accuracy is also deficient, so it is an urgent need for a location strategy suitable for mobile sensor networks.

In [9], Hu and Evans proposed a Monte Carlo-based mobile wireless sensor network node localization algorithm-MCL for node location of wireless mobile sensor networks. The algorithm proposes a simulation-based solution to estimate the posterior probability distribution of nonlinear discrete-time motion models. In view of the low efficiency of position sampling of the MCL algorithm, many improved mobile node localization algorithms have been developed based on MCL, such as Monte Carlo Localization Boxed algorithm (MCB) [10], range-based-MCL algorithm [11], RSSI-MCL algorithm [12], etc. The MCB algorithm establishes the sampling range of the anchor box limit position prediction by making full use of the one-hop (two-hop) anchor node information that the unknown node can perceive, which effectively improves the accuracy and efficiency of the node positioning [13]. However, the MCB still has a large and fuzzy sampling point set, and there is also much room for improvement in the prediction of the motion direction of the node. In addition to the above improved strategies, there are many researches on the localization algorithm of wireless mobile sensor networks based on Monte Carlo localization algorithm, such as adaptive weight [14], virtual anchor node [15], model prediction [16], fusion posture estimation [17] etc. Aiming at the problem of MCB algorithm, this paper proposes an improved MCB for mobile sensor networks. The algorithm further limits the sampling range of the nodes to be located by using the historical anchor node and its RSSI ranging, and distinguishes the weights of the effective sampling nodes by RSSI ranging of the surrounding anchors and current neighbor nodes. In addition, an improved motion model which is conducive to the prediction of the node's motion direction is proposed.

The rest of the paper is organized as follows. Section 2 discusses the related localization algorithm MCL and MCB. Section 3 introduces the proposed IMCB from the aspect of improvement ideas, optimization method of sampling weight based on RSSI and improved motion model for sampling prediction. Section 4 provides the simulation results of IMCB and analyzes the comparison to existing algorithms. Finally, section 5 concludes the paper.

## 2. Related Work

### 2.1. MCL Algorithm

MCL algorithm has the following assumptions: all nodes are movable and the time is divided into several discrete time slots with equal length, and the node position is updated once in each time unit; the unknown node only knows its maximum moving distance $v_{\max}$ in unit time; the communication range of each node is *r*.

In the following description, $l_t$ represents the position distribution of common nodes at time $t$. $o_t$ represents the observation value sent by anchor nodes from $t-1$ to $t$. $L_t = \{l_t^0, l_t^1, \mathrm{L}\ , l_t^{48}, l_t^{N-1}\}$ represents the possible position sampling set of nodes at time *t*, including N samples. $p(l_t \mid l_{t-1})$ represents the prior probability of unknown node predicting the position at current time based on the previous time. $p(l_t \mid o_t)$ represents the posterior probability of $l_t$ obtained based on the observation value $o_t$. MCL algorithm is divided into four phases: initialization, prediction, filtering and locating.

(1) Initialization: Each common node constructs a sampling set of its own possible positions $L_t = \{l_t^0, l_t^1, \mathrm{L}\ , l_t^{48}, l_t^{N-1}\}$.

(2) Prediction: The common node extracts a new sampling set $L_t$ from the position information $L_{t-1}$ and motion information of the previous moment. Then the possible node position at the current time is in the circular area $C_v$ with the center of $l_{t-1}^i$ and the radius of $v_{\max}$. The samples in the circle are uniformly distributed:

$$p(l_t \mid l_{t-1}) = \begin{cases} \dfrac{1}{\pi v_{\max}^2} & d(l_t, l_{t-1}) < v_{\max} \\ 0 & d(l_t, l_{t-1}) \ge v_{\max} \end{cases} \tag{1}$$

(3) Filtering: At this phase, according to the observation values of one hop and two hop anchor nodes received, common node filters the invalid position samples in $L_t$, the one-hop neighbor node receiving the message should be in the circle with the anchor node as the center and *r* as the radius, while the two-hop neighbor node should be in the circle with the anchor node as the center and the radius of *(r, 2r)*, and the position sampling that does not meet the condition is invalid sampling, that is:

$$p(l_t \mid o_t) = \begin{cases} 1 & \forall s \in S, d(l,s) \le \mathrm{r} \wedge \forall s \in T, \mathrm{r} < d(l,s) \le 2\mathrm{r} \\ 0 & \qquad\qquad\qquad\qquad\qquad\qquad \textit{others} \end{cases} \tag{2}$$

Where $s$ represents an anchor node, $S$ represents a set of one-hop anchor nodes, and $T$ represents a set of two-hop anchor nodes. In order to obtain enough position samples, when the invalid samples in the sample set are filtered out, the previous prediction and filtering process are repeated until the number of samples is met or the upper limit of sampling rounds is reached.

(4) Locating: Finally, the estimated position $es\_p_t$ of the common node is the average value of the sample set, where the weight $w_t^i$ of the sample point is equal to 0 or 1.

$$es\_p_t = \sum_i \frac{l_t^i w_t^i}{\sum_i w_t^i} \tag{3}$$

## 2.2. MCB Algorithm

MCB algorithm is an improved method to solve the problem of low efficiency and long computation time caused by the ambiguity of sampling range of MCL algorithm, that is, the communication range of anchor nodes is used to constrain sampling area. In the MCB algorithm, for convenience, a circular communication domain with the anchor node *S* as the center and the communication distance r as the radius is approximated as a circumscribed square of the circle with the anchor node S as the center and *2r* as the side length, called the anchor box. If a two-hop anchor node is used, the side of the anchor box is *4r*. The other phases of the MCB algorithm are the same as the MCL algorithm, but the anchor node sampling box is established during the sampling phase, as shown in Fig.1. Sampling in the overlapping area of these anchor boxes can effectively narrow the sampling range, improve the sampling efficiency and quality, shorten the positioning time, reduce the calculation amount, and improve the positioning speed and accuracy.

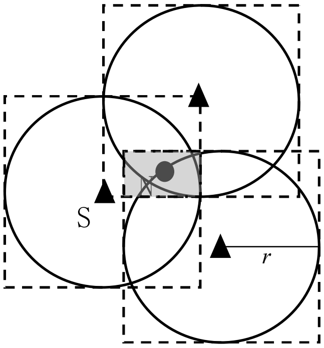


**Fig.1.** Sampling in the overlapping area of the anchor boxes can effectively narrow the sampling range

The anchor node sampling box is a circumscribed square overlapping area in which the anchor node *S* is the center of the circle and the communication range *r* is a radius. This area can be expressed as follows:

$$\begin{cases} x_{\min} = \max_{j=1}^{n}(x_j - r) \\ x_{\max} = \min_{j=1}^{n}(x_j + r) \\ y_{\min} = \max_{j=1}^{n}(y_j - r) \\ y_{\max} = \min_{j=1}^{n}(y_j + r) \end{cases} \tag{4}$$

Where $x_{min}$, $x_{max}$, $y_{min}$ and $y_{max}$ are the coordinate range of the sampling box, *n* is the number of anchor nodes, and $x_j$, $y_j$ are the horizontal and vertical coordinates of the *j-th* anchor point. Fig. 1 and (4) show the case of a one-hop anchor node. If a two-hop anchor node is used, in the above figure and (4), $r = 2r$. The velocity sampling box at the previous moment is constructed in the same way as the MCL algorithm, and then the intersection of the two boxes is sought.

# 3. An Improved MCB Algorithm Based on weighted RSSI and motion prediction

## 3.1. Improved MCB Algorithm

Although, the MCB algorithm limits the sampling range of position prediction by making full use of the information of one-hop (two-hop) anchor nodes that unknown nodes can "hear", and effectively improves the accuracy and efficiency of node positioning [10]. However, there are still some problems in MCB: firstly, MCB can get the final estimated position by averaging all sampling points, without effectively distinguishing good sampling points from bad ones; secondly, the motion models used in MCB and MCL algorithm are all improved Random Waypoint Mobility Models, the movement direction of nodes in each time unit are arbitrarily selected. However, in practice, the movement direction range of nodes should be limited.

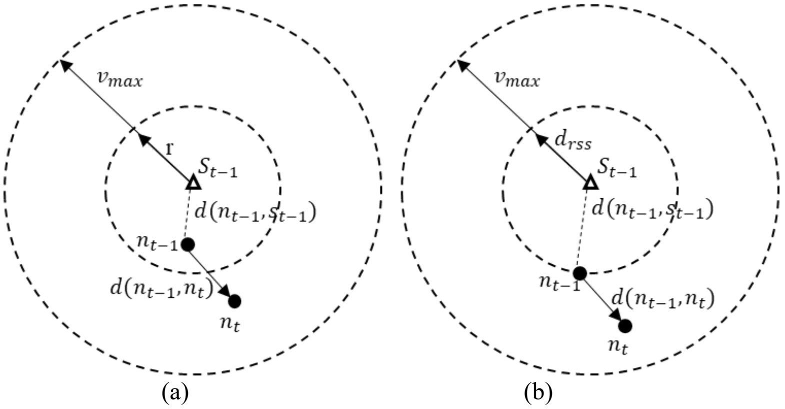
In order to further improve the sampling efficiency of MCB algorithm and solve the above two problems, this paper proposes the scheme which can further reduce the sampling area of unknown nodes by using the historical anchor nodes, optimize the weight of sample points in the sampling set by using the received signal strength indication information between the unknown nodes and the neighbor nodes, then dynamically predict the movement direction of nodes in the sampling phase.

Suppose that the anchor box has been established according to the steps described above, and the sampling area of the node position is reduced according to the anchor node information of the node to be located at time $t-1$, so as to further improve the sampling efficiency. As shown in Fig. 2 (a), $n_{t-1}$ is the position of the node to be located at time $t-1$, $\mathrm{n_t}$ is the position of the node to be located at time $t$, and $s_{t-1}$ is the neighbor anchor node of the node to be located at time $t-1$. Because $n_{t-1}$ must be in the communication range of $s_{t-1}$, that is, $d(n_{t-1},s_{t-1})$ must be less than *r*, and the maximum moving distance of $n_t$ is $v_{\max}$, so $n_t$ must be in the circle with $s_{t-1}$ as the center and $r + v_{\max}$ as the radius.

As shown in Fig. 2 (b), the distance $d_{rss}$ from $n_{t-1}$ to $s_{t-1}$ can be calculated by the received signal strength. According to the above principle, $n_t$ can be limited to a circle with $s_{t-1}$ as the center and $d_{rss} + v_{\max}$ as the radius, further reducing the limit range of $s_{t-1}$ to $n_{t-1}$. In this way, with reference to the method of MCB, the circumscribed square of the circle is established. $X(s_{t-1}, j), Y(s_{t-1}, j)$ are the abscissa and ordinate of the *j-th* anchor node at time $t-1$, respectively. Then the overlap between the circle and the anchor box can be calculated as follows:

$$\begin{aligned} x_{\min} &= \max_{j=1}^{m}\{x_{\min}, X(s_{t-1}, j) - d_{rss} - v_{\max}\}, x_{\max} = \max_{j=1}^{m}\{x_{\max}, X(s_{t-1}, j) + d_{rss} + v_{\max}\} \\ y_{\min} &= \max_{j=1}^{m}\{y_{\min}, Y(s_{t-1}, j) - d_{rss} - v_{\max}\}, y_{\max} = \max_{j=1}^{m}\{y_{\max}, Y(s_{t-1}, j) + d_{rss} + v_{\max}\} \end{aligned} \tag{5}$$

After the sampling box has been established as above, position sampling can be carried out for the nodes to be located in the sampling box.



(a) (b)

**Fig.2.** In order to further constrain the node position sampling area, the historical anchor node information and RSSI ranging results are used to limit the position sampling at the current time

In the filtering phase, common nodes filter the invalid position samples in $L_t$ according to the observation values of one-hop and two-hop anchor nodes received. The basic method is the same as that of MCL algorithm, and then the restriction conditions of historical anchor nodes are added, namely:

$$p(l_t \mid o_t) = \begin{cases} 1 & \forall s \in S, d(l,s) \le r \wedge \forall s \in T, r < d(l,s) \le 2r, \\ & \wedge \forall s \in H, d(l,s) \le r + v_{\max} \wedge Meet\ motion\ model\ limitations \\ 0 & \qquad\qquad\qquad\qquad others \end{cases} \tag{6}$$

Where, $l$ represents the position sampling point, $s$ represents an anchor node, $S$ represents a set of anchor nodes, $T$ represents a set of two anchor nodes, and $H$ represents a set of historical anchor nodes. For the limitations of the motion model on the sampling points, see section 3.3. In order to obtain enough position samples, when the invalid samples in the sample set are filtered out, the previous predictive sampling and filtering process are repeated until the number of samples is met or the upper limit of sampling rounds is reached.

### 3.2. Optimization of sampling weight based on RSSI

In the actual wireless communication, the surrounding environment is very complex. The reflection, diffraction and scattering of electromagnetic wave will cause the signal fading. In the system simulation, the lognormal shadow fading model is often used to characterize the channel characteristics, and the formula is as follows:

$$P_r(d) = P_r(d_0) - 10n\lg\frac{d}{d_0} + X_\sigma$$

(7)

Where, $P_r(d)$ is the received signal strength at the distance $d$, $P_r(d_0)$ is the received signal strength at the reference distance $d_0$, and $n$ is the path loss exponent, which represents the path loss rate. In free space, $n$=2. $X_\sigma$ is a random variable with Gaussian distribution of zero mean, and its standard deviation is $\sigma$ in dB. According to formula 7, the propagation distance of the signal can be calculated according to the received signal strength of the node, and the calculated value can be used to approximate the real distance between nodes.

In this paper, RSSI ranging is used to estimate the distance between the node to be located and its one hop anchor node, which is stored by the node to be located. When positioning, it is necessary to obtain the distance $d_{rss}$ between the node to be located and its historical anchor node at $t$-1 time. If the distance between node $i$ to be located and one hop anchor node $j$ obtained by RSSI ranging is $\hat{d}_{ij}$, for practical consideration, this distance cannot exceed the communication range, and the its value can be estimated as:

$$d_{\text{rss}} = \min(\hat{d}_{ij}, r)$$

(8)

This distance will be used to limit the position sampling area of the node to be located.

The position estimation based on RSSI weight can be divided into two stages: the non-weighted position estimation and the weighted position estimation. In the first stage, the common node obtains the average position estimation according to the filtered effective sampling points, and sends the value and its position error estimation to the common neighbor node; in the second stage, the common node calculates the weight through the RSSI distance according to the neighbor anchor node information, the neighbor node's position estimation and its position error estimation, and updates the estimated position. The RSSI weights are calculated as follows:

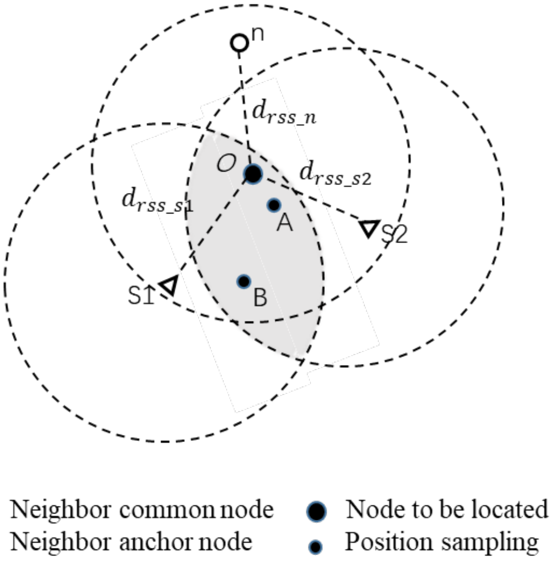


**Fig.3.** Computing sample weights according to the neighbor nodes' RSSI

As shown in Fig.3, *O* is the node to be located, *S1*, *S2* and *n* are two anchor nodes and one common node within the communication range of the node to be located. The intersection of circle *S1* and circle *S2*, namely the shadow part, is the effective range of *O* position sampling, and *A* and *B* are the two position sampling within the effective range. The distance between the unknown node and the anchor node and the neighbor node can be estimated by the signal RSSI received by the unknown node, which is labeled $d_{rss\_s1}$, $d_{rss\_s2}$ and $d_{rss\_n}$.

The distance from the sampling point to the anchor node can be calculated according to the known coordinates. The distance from the sampling point to the common neighbor node can be approximated by the non-weighted estimated distance from the sampling point to the neighbor node, marked as $d(l_t^i,s1)$, $d(l_t^i,s2)$ and $d(l_t^i,n)$. Considering the errors of RSSI ranging and neighbor node position estimation, RSSI ranging error factor $\delta_{rssi}$ ($\delta_{rssi}>0$) and neighbor node position error estimation $ER_n$ are added to the weight calculation. The weight calculation method is: if $l_t^i$ satisfies three inequalities in equation (9) at the same time, then the weight $\omega_t^i$ of $l_t^i$ plus 1. For example, the weight of position sampling *A* in Fig.3 is greater than that of *B*.

$$\begin{cases} \left|d_{rss\_s1}-d(l_t^i,s1)\right|<\delta_{rssi} \\ \left|d_{rss\_s2}-d(l_t^i,s2)\right|<\delta_{rssi} \\ \left|d_{rss\_n}-d(l_t^i,n)\right|<ER_n+\delta_{rssi} \end{cases}$$

(9)

The computing method of $ER_n$ is as follows: Establish a minimum rectangular box for the filtered effective sampling points, so that all the effective sampling points can be included, as shown in Fig.4. Point $es\_p$ is the estimated position of all the effective position sampling points after the non-weighted average. The variables $dx$ and $dy$ are the maximum errors of the sampling points in X and Y directions, then:

$$ER_n = \sqrt{dx^2 + dy^2} \tag{10}$$

1. When there are three or more anchor nodes in the range of the unknown node, only anchor nodes are taken for the weight computing of position sampling;
2. If the number of anchor nodes in the communication range of the unknown node is less than three, the neighbor nodes with the smallest $ER_n$ are used to supplement the number, and then neighbor anchor nodes and common nodes are used for the weight computing of position sampling;
3. If the total number of neighbor anchor nodes and common nodes is less than three, no weight computing will be carried out, and the result of non-weighted position estimation in the first stage will be taken as the position estimation of unknown nodes.

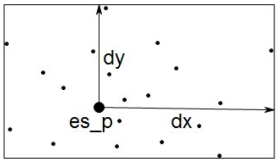


**Fig.4.** Computing the Estimation of Localization Error, where black dots represent all valid position samples of a neighbor's common node

## 3.3. Optimization of sampling prediction based on motion model

The basic idea of the sampling prediction optimization is that the motion model restricts the motion direction of the nodes, and the prediction sampling points of the unknown nodes should also be limited within the range of the possible motion direction.

### Improvement of motion model

The motion model used in MCL algorithm simulation is an improved random middle point motion model, which takes the current position of the node as the starting position, selects any position within the deployment area of the node as the destination position to determine the movement direction of the node, and uses any value within the range (minimum rate, maximum rate) as the movement rate of the node to move towards the destination position. The algorithm considers that the motion in different time slots is independent of each other, which may cause some unrealistic motion behaviors of nodes, such as sharp turning 180 degrees. Therefore, this paper improves the motion model of the node on the basis of the random middle node motion model, and limits the motion direction of the node to a realizable range, that is, the angle between the motion direction of the node selected at the next moment and the original motion direction should be less than the maximum realizable angle. As shown in Fig.5, $l_{t-2}$ and $l_{t-1}$ are the positions of the nodes in the first two moments respectively, and $l_t$ is the position where the nodes are about to arrive. $\alpha$ is the angle between vector $\overset{\text{uuuu}}{\boldsymbol{l}_{t-1}\boldsymbol{l}_t}$ a and $\overset{\text{uuuuuu}}{\boldsymbol{l}_{t-2}\boldsymbol{l}_{t-1}}$ B, which represents the steering angle of node motion, and $\Delta\varphi$ is the maximum value of $\alpha$. Therefore, $\alpha$ is limited to $\Delta\varphi$, which avoids the unrealistic too large steering angle of the node in the process of motion.

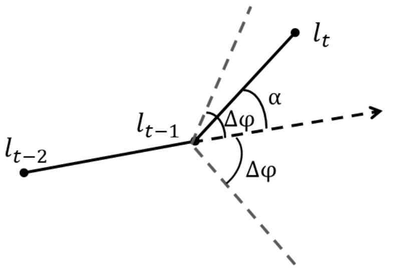


**Fig.5.** Modifying motion model which can limit the motion direction of the node by setting the maximum angle α .

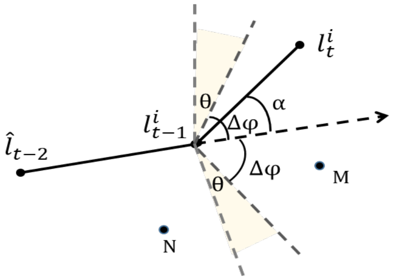


**Fig.6.** Predicting direction based on motion model and filtering the invalid nodes.

### Optimization of sampling point prediction based on motion model

On the basis of the above improved motion model, this paper predicts the dynamic motion direction of nodes based on the number of anchor nodes in the sampling phase, that is, the sampling is only in the range of possible motion direction of nodes. Fig. 6 is a direction prediction model for node position prediction sampling, in which $\overset{\$}{l}_{t-2}$ is the node position estimation at time $t$ - $2$, $l_{t-1}^i$ is the *i-th* position sampling of the node at time $t$ - $1$, $l_t^i$ is the *i-th* position sampling of the node at time $t$, where $i \in [0, N-1]$, $N$ is the maximum number of samples in the unknown node position sampling set. Because $\overset{\$}{l}_{t-2}$ and $l_{t-1}^i$ are estimated information and have errors with the real positions of nodes in the first two moments, angle $\theta$, the correction angle of $\Delta\varphi$, is added to the direction prediction model. For the convenience of the following description, the vector with $\overset{\$}{l}_{t-2}$ as the starting point and $l_{t-1}^i$ as the end point is recorded as vector $\boldsymbol{a}$; the vector with $l_{t-1}^i$ as the starting point and $l_t^i$ as the end point is recorded as vector $\boldsymbol{b}$, then the angle between $\boldsymbol{a}$ and $\boldsymbol{b}$ cannot exceed the maximum steering angle $\Delta\phi$ .

Define the estimated maximum steering angle $\Delta\phi$ as the sum of the actual maximum steering angle $\Delta\varphi$ and the correction angle $\theta$ :

$$\Delta\phi = \Delta\varphi + \theta \tag{11}$$

$$\theta = \frac{1}{m_{t-2}m_{t-1}\sigma_1 + n_{t-2}n_{t-1}\sigma_2 + k} \times \delta \tag{12}$$

$$\sigma_1 + \sigma_2 = 1 \tag{13}$$

Among them, $m_{t\text{-}2}$ and $m_{t\text{-}1}$ are the number of anchor nodes in the communication range of unknown nodes at time $t$ - $2$ and $t$ - $1$ respectively, $n_{t\text{-}2}$ and $n_{t\text{-}1}$ are the number of two hop anchor nodes at time $t$ - $2$ and $t$ - $1$ respectively, $\sigma_1$ and $\sigma_2$ respectively represent the influence ability of one-hop anchor node number and two-hop anchor node number on localization error. The more the number of anchor nodes in the communication range of unknown nodes, the smaller the localization error of $\overset{\$}{l}_{t\text{-}2}$ and $l^{i}_{t\text{-}1}$ , the smaller the correction angle $\theta$ , otherwise $\theta$ is larger; k is a constant, its function is to avoid zero denominator; $\delta$ is a settable parameter with unit "degree". As shown in Fig.6, *M* is the effective sampling points and *N* is filtered points.

## 4. Simulation and analysis

In order to effectively evaluate the performance of the improved Monte Carlo localization boxed algorithm (IMCB) proposed in this paper and compare its performance with MCL and MCB, we build a network simulation platform [9]. The platform uses JAVA as the development environment, and the simulation parameters are set as follows:

The entire simulation area is a square area A of *500m×500m,* and 320 nodes generated by each simulation are randomly distributed including several anchor nodes, and the rest are common nodes. The node moves randomly in the deployment area according to the improved motion model as above, and the actual maximum steering angle is $\Delta\varphi = 60^{\circ}$.

In the direction prediction, set $\sigma_1 = 0.8$, $\sigma_2 = 0.2$, $k = 1$ and $\delta = 120^{\circ}$, that is, when $m_{t-2}m_{t-1}\sigma_1 + n_{t-2}n_{t-1}\sigma_2 = 0$, the maximum steering angle $\Delta\phi = \Delta\varphi + \theta = 180^{\circ}$ is estimated because the accuracy of the reference direction is too low. The $v_{\max}$ is the maximum moving speed of the node, which represents the maximum moving distance of the node in each time unit. When the motion model is adopted, the node is randomly selected from $(0, v_{\max})$, $s_d$ is the density of anchor nodes, which represents the average number of anchor nodes in the range of one hop communication.

$$s_d = \frac{anchor\_num}{A} \times \pi \mathrm{r}^2 \tag{14}$$

Where *anchor_num* is the total number of anchor nodes in the deployment area, *A* is the total area of the deployment area, and *r* is the communication radius of the node.

### 4.1. Localization error

The localization error of the node is denoted by the distance between the estimated position coordinate of the node and the real position coordinate. The computing method is as follows:

$$error = \frac{\sum_{i=0}^{num} \sqrt{(x_i' - x_i)^2 + (y_i' - y_i)^2}}{num \times r} \tag{15}$$

Where $(x_i, y_i)$ is the real coordinate of node *i*, $(x_i', y_i')$ is the node location coordinate computed by the location algorithm, *num* is the number of deployed common nodes, and *r* is the communication radius of the node.

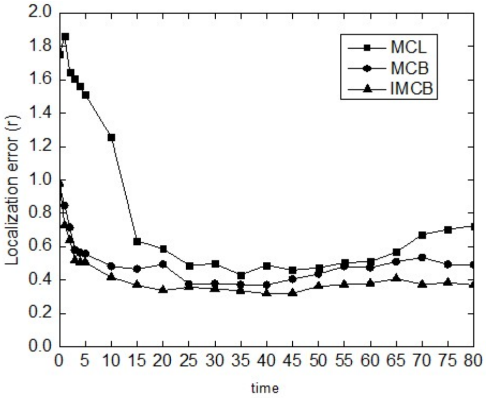


**Fig.7.** Localization error over time when $v_{max}$ = *0.2r* , $S_d = 1$

Fig.7 shows the curve of node localization error over time when $v_{max}$ = 0.2$r$ and $S_d$ = 1. The localization of nodes can be divided into initialization stage and stabilization stage. In the initialization stage, the localization accuracy of all three algorithms is very poor. With the increase of time, the localization error decreases rapidly, and the IMCB and MCB algorithm reach the stable stage faster than MCL, and the localization error of IMCB is improved obviously. In addition, after the *45-th* time unit, the localization error shows an upward trend. Due to the accumulation of errors, MCL is easy to lead to the insufficient number of node positioning prediction samples, and the localization error continues to increase. On the contrary, due to the limitation of anchor box in position prediction, MCB and IMCB prevent the worse of node localization error. IMCB makes use of the limitation of historical anchor nodes and predicts the movement direction of the nodes, which further reduces the sampling range.

At the same time, RSSI weight optimization effectively distinguishes the possibility that the sample points are close to the real position, thus obtaining more accurate location accuracy than MCB. From the simulation results, the localization error of IMCB is 11.2% lower than that of MCB. If only the value of stabilization stage is estimated, the localization error of IMCB is 12.41% lower than that of MCB.

## 4.2. Impact of maximum rate

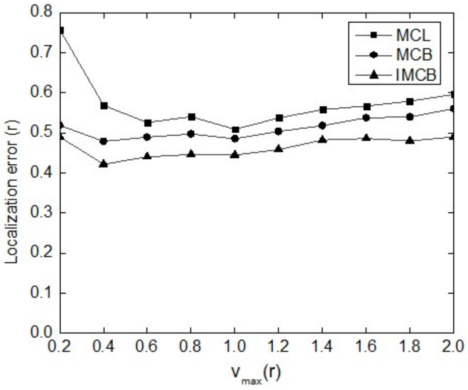


**Fig.8.** Impact of maximum rate when $s_d = 1$

The increase of $v_{max}$ will make the sampling area larger, but from another point of view, the rapid movement of nodes may bring more anchor node information. Fig.8 shows the curve of node localization error with node movement rate when $s_d = 1$. The error value is the average error after multiple localization of the specified $v_{max}$. It can be seen that with the increase of $v_{max}$, the localization error decreases significantly between *0.2r* and *0.4r*. After that, the localization errors of MCB and MCL have an obvious upward trend. Although the localization errors of the IMCB algorithm is also increase, the trend is obviously slow. This is because the sampling range of MCB and MCL position prediction will increase with the increase of $v_{max}$. But for IMCB, in addition to the limitation of anchor box and $v_{max}$, direction prediction is also added, which is independent of $v_{max}$. When $v_{max}$ increases to the extent that it is ineffective to predict the sampling range of the limited position, that is to say, when the square constructed according to $v_{max}$ completely covers the anchor box, the range limitation impact of the direction prediction is still effective. Therefore, with the increase of $v_{max}$, the rising trend of IMCB localization error is smooth, and the location accuracy is better than that of MCB.

### 4.3. Impact of anchor node density

Increasing the density of anchor nodes is beneficial for reducing the localization error, but it will undoubtedly increase the deployment cost of the network. Fig.9 shows the curve of localization error changing with the density of anchor nodes when $v_{\max}=0.2r$. It can be seen from the figure that the localization error of IMCB is lower than that of MCB. On the one hand, the increase of the number of anchor nodes makes the weight optimization based on RSSI more effective. On the other hand, the increase of anchor nodes improves the location accuracy, and also makes the range of direction prediction angle smaller in the next positioning, thus reducing the range of position prediction and further improving the location accuracy. In addition, the constraints of historical anchor nodes make the location accuracy of IMCB due to MCL and MCB when the density of anchor nodes is low.

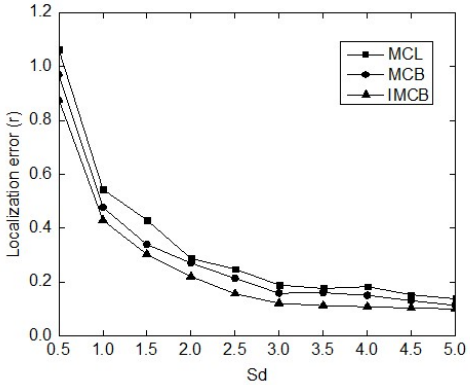


**Fig.9.** Impact of anchor node density where $v_{\max} = 0.2r$

## 5. Conclusion

Location problem is one of the hotspots in wireless sensor network research. In order to further improve the performance of the MCL algorithm which has attracted much attention in the field of mobile sensor network in recent years, this paper designs a weighted localization algorithm named Improved Monte Carlo Localization Boxed (IMCB) based on the historical anchor node and RSSI ranging, which has the following advantages: firstly, using the historical anchor node and the historical RSSI ranging information to further narrow the sampling range of the unknown node, improve the sampling efficiency of the node position, and also to some extent alleviate the problem of anchor node density; Secondly, the weight optimization based on RSSI effectively distinguishes the weight of the sampling points, which is conducive to further reducing the localization error of the nodes; thirdly, the improvement of the motion model is beneficial to the direction prediction of the nodes, reducing the sampling range of the position prediction, improving the localization efficiency and accuracy of the nodes. However, there are still many factors affecting the location algorithm, such as computational complexity, location time, environmental disturbance, network security, etc. which will affect the performance and conditions for use of the algorithm. More in-depth research and evaluation are needed.

**Acknowledgment.** The authors wish to acknowledge the Beijing Municipal Natural Science Foundation of China (No. 4172045) for the support in funding this work.

## References

1. W. Cui, S. C. Wu, Y. Z. Wang. A Gossip-Based AOA Distributed Localization Algorithm for Wireless Sensor Networks [J]. Applied Mechanics and Materials, 2014, Vol. 577: pp. 841-846.
2. M. Jamalabdollahi and S. Zekavat. Time of Arrival Estimation in Wireless Sensor Networks via OFDMA [C] // Vehicular Technology Conference. Boston, MA: IEEE, 2015, pp. 1-5.
3. Weile Zhang , Qinye Yin, Wenjie Wang. Distributed TDoA estimation for wireless sensor networks[C] // Proceedings 2010 IEEE International Conference on Acoustics, Speech and Signal Processing: 2010, p 2862-5.
4. F. Yaghoubi, A. A. Abbasfar and B. Maham. Energy-Efficient RSSI-Based Localization for Wireless Sensor Networks [J]. IEEE Communications Letters, June 2014, vol. 18, no. 6, pp. 973-976.
5. Laurendeau C, Barbeau M. Centroid Localization of Uncooperative Nodes in Wireless Networks Using a Relative Span Weighting Method[J]. Eurasip Journal on Wireless Communications & Networking, 2010, 2010(1):1-10.
6. Agashe A A, Agashe A A, Patil R S. Evaluation of DV Hop Localization Algorithm in Wireless Sensor Networks[C]// International Conference on Advances in Mobile Network, Communication and ITS Applications. 2012:79-82.
7. P. Ren, W. Liu, D. Sun and K. Liu. Node Localization based on Convex Optimization in Wireless Sensor Networks [C]// Fuzzy Systems and Knowledge Discovery (FSKD), 12th International Conference. Zhangjiajie: IEEE, 2015: 2169-2173.
8. Zhang Aiqing, Ye Xinrong, Hu Haifeng. Point in triangle testing based trilateration localization algorithm in wireless sensor networks [J]. Internet and Information Systems, 2012, vol.6: 2567-2586.
9. Hu L, Evans D. Localization for mobile sensor networks [J]. Wireless Communication, 2004:45--57.
10. Aline Baggio , Koen Langendoen . Monte-Carlo Localization for Mobile Wireless Sensor Networks . Lecture Notes in Computer Science , 2006 , 4325 (1 1):317-328.
11. Dil B,Dulman S,Havinga P. Range-based localization in mobile sensor networks [J]. Wireless Sensor Networks, 2006: 164-179.
12. Wang W D, Zhu Q X. RSS-based Monte Carlo localisation for mobile sensor networks [J]. Communications, IET, 2008, 2(5): 673-681.
13. Baggio and K. Langendoen. Monte-Carlo Localization for Mobile Wireless Sensor Networks [C]// Mobile Ad-Hoc and Sensor Networks , 2006: 317-328.
14. P. Wang, N. Chen, F. Zha, W. Guo and M. Li, "Research on Adaptive Monte Carlo Location Algorithm Aided by Ultra-Wideband Array," 2018 13th World Congress on Intelligent Control and Automation (WCICA), Changsha, China, 2018, pp. 566-571.

15. J. Luan, R. Zhang, B. Zhang and L. Cui. An Improved Monte Carlo Localization Algorithm for Mobile Wireless Sensor Networks [C]// Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium. 2014: 477-480.
16. P. Li, Y. Zhang and Q. Zhang, "Monte Carlo Location Algorithm Based on Model Prediction," 2018 International Symposium in Sensing and Instrumentation in IoT Era (ISSI), Shanghai, 2018, pp. 1-5.
17. W. Chen, T. Huang and A. Maalla, "Research on Adaptive Monte Carlo Location Method Based on Fusion Posture Estimation," 2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 2019, pp. 1209-1213.

**Chunyue Zhou** completed her BSc in Information Engineering, MSc and PhD in Communication &Information Systems from the School of Electronic and Information Engineering, University of Beijing Jiaotong, China. She has been working in the areas of network technology and security for many years. She is now a Senior Engineer and her research interests include the next generation Internet and information security.

**Hui Tian** is a Senior Lecturer and the Discipline Head of Computer Science in Griffith University. She has been working in the areas of network analysis and optimization, privacy and security for over 15 years. Her contributions to research range from network topology discovery, routing protocols in wireless sensor networks to the challenging issues about data publishing in social networks with privacy preservation, anomaly detection in network traffic analysis and other security issues in networks.

**Baitong Zhong** completed his BSc in Electronic & Information Engineering from the Northeast Petroleum University, MSc in Information Technology from the School of Computer Science and Information Technology, University of RMIT, Australia. He is now a teaching assistant in Hunan Electronic Technology Vocational College and his research interests include the IOT technology and information security.

*Received: February 04, 2020; Accepted: July 15, 2020*

Computer Science and Information Systems 17(3):795–817 https://doi.org/10.2298/CSIS190903021Y

# VDRF: Sensing the Defect Information to Risk Level of Vehicle Recall based on Bert Communication Model

Xindong You[1], Jiangwei Ma[1], Yuwen Zhang[1], Xueqiang Lv[1,*], and Junmei Han[2]

[1] Beijing Key Laboratory of Internet Culture and Digital Dissemination Research,
Beijing Information Science & Technology University, Beijing, 100101, China
[2] Laboratory of Complex systems, Institute of Systems Engineering. AMS. PLA, Beijing,
100029 China
*Corresponding Author lxq@bistu.edu.cn

**Abstract.** The recall of defective automobile products is one of the important measures to promote the quality of product quality and protect consumers' pyhsical safety and property security. In order to assess the risk level of defect cases, automobile recall management experts need to analyze and discuss the defect information by personal. A risk level prediction method based on language pre-training Bert model is proposed in this paper, which can transform the defect information into rick level of the vehicle and then predict vehicle recall automatically, in which a seq2seq model is proposed to multi-label the vehicle complaint data. The outputs of the seq2seq model combined with other static and dynamic information are used as the input of the Bert communication model. Substantial comparative experiments of different feature combinations on different methods show that the proposed VDRF method achieves F1 value with 79% in vehicle recall risk prediction, which outperforms the traditional method.

**Keywords:** Bert communication model, defect information transforming, multi-label classification, risk level prediction.

## 1. Introduction

With the continuous development of the vehicle industry, vehicles have become a necessity in people's lives. A large number of consumer complaints are collected in the vehicle quality defect complaint system, which named the defect information collection system of Defective Product Administrative Center [1]. A variety of problems or failures often occur during using the automobiles. Some of these problems are caused by improper operation or other external reasons in the process of using, and have nothing to do with the automotive products themselves. Another part of the problem is usually due to negligence in the production and design process of automobiles. These products have their own design defects, because of the particularity of the automobiles, these defects will threaten the safety of consumers' lives and property to a certain extent. Defects in vehicles can cause bodily harm and sometimes fatal consequences. Moreover, defects in automotive products can have a devastating impact on the sales and reputation of automakers, especially in the social media era. In order to avoid this risk, Europe and the United States established their own defective vehicle recall system. The status quo has formed a complete recall system of automotive products. With the continuous development of China's automobile industry, the corresponding defective automobile product recall system is gradually improving. Automobile consumers are more and more inclined to use the Internet platform to release vehicle defect information. Early detection of defects not only protects consumers from economic loss, but also mitigates the financial loss of manufacturers. In the process of defective vehicle recall, defect information is an important basis to judge the risk of vehicle recall. In order to assess the risk level of defect cases, automobile recall management experts need to analyze and discuss the defect information submitted by consumers to determine whether to carry out relevant recall work, which takes a lot of time and energy. Employing natural language processing technology and in-depth learning technology to process and analyze the defect information can help defect recall managers better analyze and assess the severity of automobile defects.

In order to evaluate the severity of automobile defects, the risk level prediction of automobile defective product recall is investigated fully in this paper, and a risk prediction model based on language pre-training Bert communication model (VDRF) is proposed. The proposed VDRF communication model can sensing the defect related information into the risk level of vehicle recall automatically. Firstly, the original data is preprocessed and a data set of automobile defect cases with a certain scale is constructed. Thereafter, multi-dimensional features are extracted, such as static features, dynamic features and fault semantic features. Finally, the extracted different combinations of features are used to predict the recall risk level of the vehicle.

As a whole, the main contributions of this paper are listed as follows:

*(1) Two vehicle complaint datasets are constructed through utilizing web crawler technology, in which all kinds of complaints in the process of vehicle recall are contained.*

*(2) A Seq2seq neural network model is firstly employed to solve the multi-label classification on vehicle complaint data, in which the defect label features and defect label distribution are added to the basic seq2seq model, which makes the model more suitable for multi-label classification of vehicle complaint data.*

*(3) The pre-training language model Bert model is used to predict the risk lev-el of vehicle recall. Static feature, dynamic feature and fault semantic feature are extracted to classify the risk level, so that the semantic information in fault description can be better captured.*

*(4) Substantial comparative experiments of different feature combinations and different methods are conducted, which show that the proposed method achieves F1 value with 79% in vehicle recall risk prediction, which outperforms the traditional method.*

## 2. Relate Work

Multi-label classification of the defect information is the preorder of the risk level prediction, and is the important part of this paper. Therefore, related work of multi-label classification and risk level prediction are investigated in this section.

Multi-label classification mainly includes three types of solutions. They are problem transformation methods, algorithm adaptation methods and neural network-based methods. The idea of problem transformation is to transform multi-label problem into single-label classification problem in some way, a mature single label classification method is used to solve the problem. Binary Reliance (BR) algorithm proposed by Boutell [2] transforms each label into a single label classification problem, which is independent of each other. The disadvantage of this method is that the relationship between labels is ignored. Similar algorithms include LIFT algorithm [3], Label Powerset (LP) algorithm [4], and Classifier Chain (CC) algorithm [5].

The algorithm adapts to multi-label data after modifying and extending the traditional single-label classification algorithm. Clare [6] extends the definition of information entropy to multi-label problem, and then uses improved decision tree algorithm to classify multi-label [7]. Elisseeff [8] proposes Rank-SVM algorithm by introducing loss function to support vector machine (SVM). Zhang and Zhou [9] proposed an improved ML-KNN algorithm based on k-nearest neighbor algorithm to solve the multi-label classification problem. Li [10] proposed a new joint learning algorithm, which propagates the feedback of the current label to the classifier of the subsequent label, and achieves good results in text multi-label classification.

Neural network models are applied to multi-label learning tasks recently. Zhang and Zhou [11] proposed BP-MLL model, which uses a new loss function in the fully connected neural network. Experiments show that the neural network model can capture the characteristics of multi-label tasks. Chen [12] uses a com-bination of CNN and RNN to represent the semantic information of the text and the higher-order features between the labels. Baker [13] maps the rows of co-occurrence labels to initialize the final hidden layer of the CNN, which can improve the performance of the model. Yang [14] claimed that multi-label classification task should be regarded as sequence generation problem. They use a new sequence generation model with a new decoder structure to solve the multi-label classification problem, and achieved good results.

In the research of automobile defect recall prediction, Zhang [16] proposed a new method to predict automobile recall risk based on the content published by users in the forum. For defective vehicles, before manufacturers and government agencies take investigative action, vehicle forums on the Internet typically dis-play user-posted content containing features of a defective vehicle. Through statistical analysis, it is found that there are overlaps between these contents and the official recall notices. It is of great significance for vehicle recall work to study the use of various machine learning algorithms to predict the risk of vehicle recall using defect features. Yang Shuanglong [17] collects the complaint data of various automobile platforms on the Internet through a large number of automated ways, and uses data mining methods to carry out risk recall early warning research on automobile products. It mainly includes automatic collection and pretreatment of automobile complaint data, text classification based on automobile complaint data, and early warning of automobile recall risk based on complaint data. Jiang Cuiqing [18] and others need a lot of manual labeling for the classification process in the research of automobile defect discovery, and that the classification of defective contents according to product components is not completely applicable. Based on Chinese social media, a framework of automobile defect recognition and automobile defect feature set in constructed in this paper, studies the method of automobile product defect classification using semi-supervised learning algorithm and the subject modeling of automobile product defect using LDA, and achieves good results. There are also some researches in this field abroad. Abrabhams [19] proposed a framework for automobile product defect detection based on the relevant information published by users on auto-mobile forums. In the framework, firstly, the relevant feature information is mined by text mining technology, and then a regression model for automobile product defect detection is constructed. Abrabhams [20] constructs a text mining model that can identify the auto parts involved in the user's post content. A binary classifier according to the name of the forum sub-module published by the post as the tag of the post is constructed in this paper, which classifies the content of the forum post according to the auto parts involved [21].

## 3. VDRF: Prediction of Risk Level of Vehicle Recall based on Defect Information

### 3.1. Model Architecture of VDRF

An overview of our proposed model is Figure 1. Firstly, we construct the automobile domain dictionary and the automobile defect label library according to the data on the Internet. Then we expand the automobile defect label library by using the automobile domain dictionary, and get the synonymous description of the automobile defect label library. According to the automobile defect label library, we classify the data of automobile complaints and get the defect label. Finally, we use Bert to predict the risk level of automobile recall based on the static and dynamic features, defect labels and defect severity levels extracted from automobile defect data.

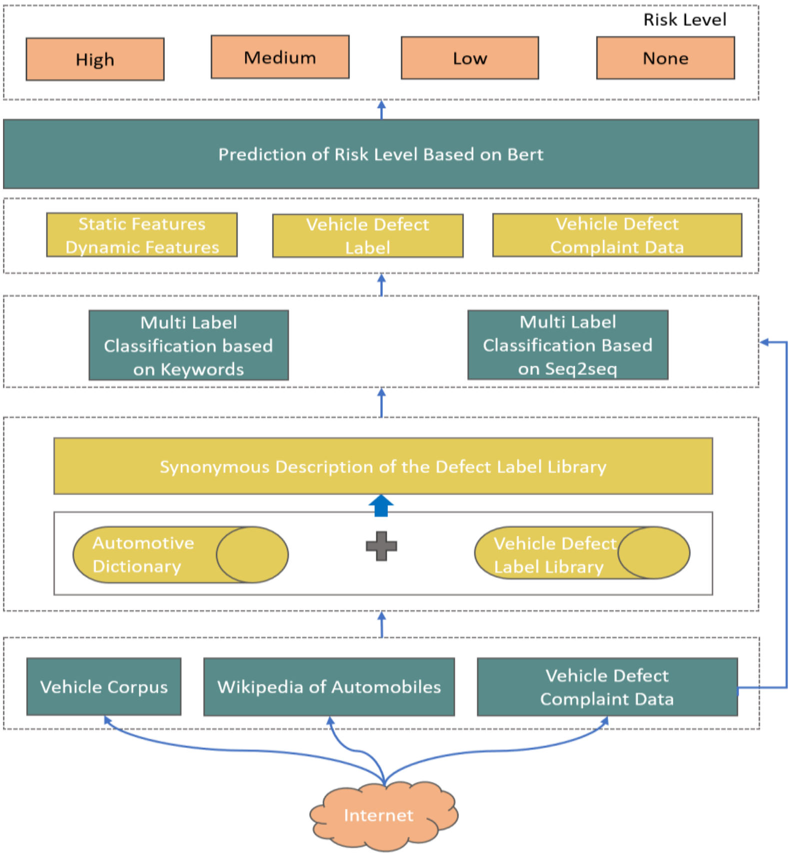

Risk Level
High
Medium
Low
None
Prediction of Risk Level Based on Bert
Static Features Dynamic Features
Vehicle Defect Label
Vehicle Defect Complaint Data
Multi Label Classification based on Keywords
Multi Label Classification Based on Seq2seq
Synonymous Description of the Defect Label Library
Automotive Dictionary
Vehicle Defect Label Library
Vehicle Corpus
Wikipedia of Automobiles
Vehicle Defect Complaint Data
Internet

**Fig. 1.** Architecture of VDRF Model

### 3.2. Defect Label Library Feature

The vehicle defect label library consists of standardized vehicle defect names and corresponding typical defect descriptions. Embedding layer of the model used in this paper includes two parts, one part is based on the word vector. And the other part reflects whether the key words in the defect description appear corresponding vehicle defect description directly. Considering that the complaint data are from different kinds of consumers of different cultural levels, different descriptions may appear for the same group of different users of the defect, we expanded the synonym of the existing defect label library in this part. After analysis, the defect description is usually composed of secondary assembly and specific defect description, such as "door rust". The secondary assembly is mainly the name of the vehicle parts. We extend the nickname, abbreviation and common misnomer of vehicle parts by search engine. For the vehicle defect description part, we use the synonym extension tool synonyms [22] to extend this collection. We replace the word vector model of the toolkit with the pre-trained vehicle domain word vector. Candidate words are selected by similarity of defect description.

Finally, a defect label library with extended synonymous descriptions is obtained. In the embedding layer of the model, the representation of a word is divided into two parts, one is the word vector represented by the domain word vector model, and the other is the 32-bit defect coding feature bits trans-formed from the defect coding. For each word in the complaint text, if the current word belongs to the defect label library or the corresponding secondary assembly appears in the text, the word defect coding feature position of the complaint text is defect code, otherwise the defect coding bit of the word is ‘0000’.

**Table 1.** Vehicle defect label library code

| First assembly | Second Assembly | Defect Label | Defect Code |
|---|---|---|---|
| 车身<br>car body | 车门<br>doors | 车门生锈<br>Rusting of doors | 5002 |
| 车身<br>car body | 车门<br>doors | 车门缝隙<br>doors gap | 5007 |
| 发动机<br>engine | 进排气系统<br>Intake and exhaust | 排气管脱落<br>pipes fall off | 2104 |
| 发动机<br>engine | 点火与起动系统<br>starting system | 喷油嘴故障<br>Injector fault | 2205 |
| 制动系统<br>brake | 制动通用装置<br>brake device | 回位不良<br>return fault | 6310 |

### 3.3. Multi-label Classification of Vehicle Defect Information Collection based on Seq2seq Model

The basic idea of seq2seq is using Bi-LSTM as encoder to read the input sentence, that is, the whole sentence is compressed into a fixed dimension of the code, and then use another LSTM called decoder to read the code, the information of the sentence will be compressed into a vector. And the architecture of the multi-label classification of vehicle defect information is shown in Figure 2.

*Embedding*. Firstly,Word segmentation tool jieba [23] with the vehicle domain dictionary constructed in our previous published paper [24] is employed on the complaint text S. Then, the segmented complaint text S is vectorized in the embedding layer, which can reduce the input dimension and reduce the number of parameters of the neural network. Furthermore, the dense vector representation of the word vector layer can contain more semantic information [25].

*Encoder layer*. Bidirectional LSTM [26](Bi-LSTM) recurrent neural network is used to read the text information in order from the front and back two directions, and to calculate the hidden layer vector $h_i$ for each word $w$ in the complaint text S. Each word corresponds to the hidden state vector $h$, which includes the state vectors in the two directions $\vec{h}_i$. And $\overleftarrow{h}_i$ representing the semantic information centered on $i^{th}$ word.

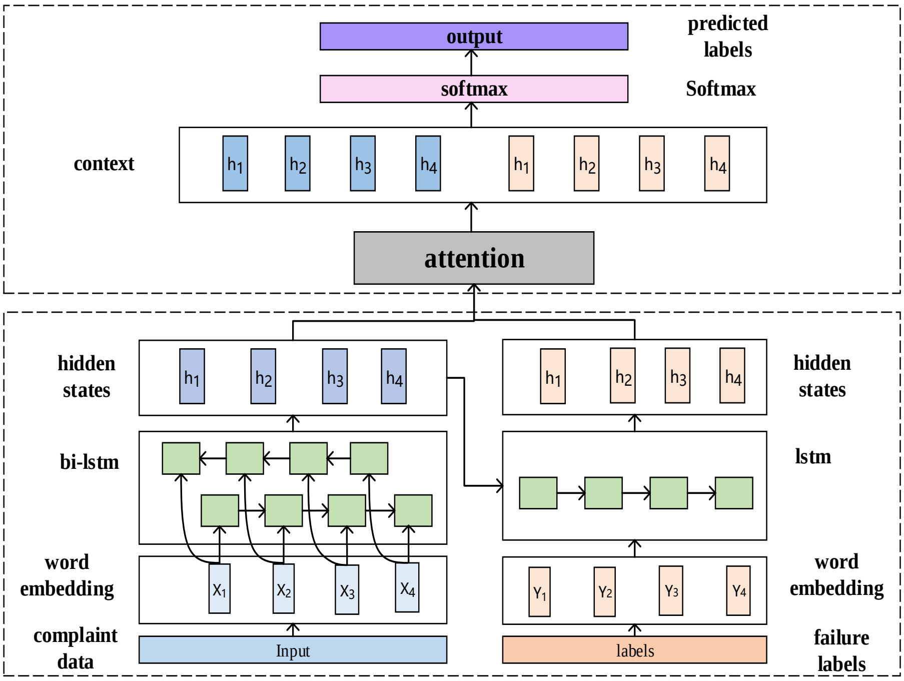


**Fig. 2.** Architecture of Multi-label Classification of Vehicle Defect Information

*Attention mechanism.* Due to the different words have different effects on prediction labels, seq2seq model with attention mechanism is used to find out the hidden state of encoder and decoder through attention connection.

*Decoder layer.* LSTM recurrent neural network is used in decoder layer. The decoder receives the hidden layer state $s_{t-1}$ at time-step $t$, the context vector $c_{t-1}$and the label distribution vector $l(y_{t-1})$ from the attention mechanism, respectively, and inputs them to the decoder. The vector $l(y_{t-1})$ reflects the overall distribution of labels. Vector $l(y_{t-1})$ is added to the decoding process can integrate the relationship between labels.

*Softmax layer.* Softmax is used as in the classification lyaer, and a defect label $y_t$ with the highest probability is generated by the output state vector $s_t$ from the decoder.

### 3.4. Vehicle Defect Recall Risk Rating Forecast Model based on Bert Model

Automobile complaint information derived from the defect information collection system of the Defective Product Management Center is used as the data source, and the automobile defect risk recall risk data set contains about 120,000 pieces of defect information. After sorting out these pieces of defect information, we finally form 10,351 typical automobile defect cases.

By analyzing the defect information data source, the main composition of defect information is shown as in Table 2.

**Table 2.** Defect information composition

| Item number | Item | Data sample |
|---|---|---|
| 1 | 缺陷信息编号<br>Defect information number | QC201312001 |
| 2 | 时间<br>Time | 201312 |
| 3 | 缺陷信息来源<br>Defect information source | 备案<br>Put on record |
| 4 | 生产者<br>Producer | 阿斯顿马丁拉共达（中国）汽车销售有限公司<br>Aston Martin Lagonda (China) Automobile Distribution Co. Ltd |
| 5 | 品牌<br>Brand | Aston Martin(阿斯顿马丁)<br>Aston Martin |
| 6 | 车型<br>Vehicle type | V8 WANTAGE |
| 7 | 里程信息<br>Mileage information | 1.38 万公里<br>13800 km |
| 8 | 使用年限<br>Service life | 2.00 年<br>2 years |
| 9 | 总成<br>Assembly | 发动机<br>Engine |
| 10 | 分总成<br>Sub assembly | 汽油发动机<br>Petrol engine |
| 11 | 故障标签<br>Defect label | 离合器液压软管夹失效<br>Clutch hydraulic hose clamp failure |
| 12 | 缺陷描述<br>Defect description | 离合器液压软管夹失效可能导致油泄漏和离合器失效，需更换新型管夹<br>Failure of clutch hydraulic hose clamp may cause oil leakage and clutch failure, so new clamp shall be replaced |
| 13 | 故障等级<br>Fault level | 中<br>Medium |
| 14 | 舆情信息影响力<br>Influence of public opinion information | 0 |
| 15 | 投诉数量 | 10 |

| | | |
|---|---|---|
| | Number of complaints | |
| 16 | 召回风险（信息会商结果）<br>Recall risk | 低<br>Low |

Defect information is mainly composed of three parts. The first part is the basic information of automobile, such as manufacturer, brand, model, mileage information and so on. The second part is the defect information of automobile fault, including the assembly and sub-assembly, and the third chapter is the result of multi-label classification of complaint information based on the defect management center automobile fault classification system, in addition, there is a defect fault severity evaluation level.

Public opinion information refers to the daily monitoring of public opinion information based on the quality and safety of automotive products and special monitoring of public opinion for specific events, mainly including public network media and We Media two sources.

In the process of defect recall, the result of information consultation is a pre-liminary judgment of recall risk made by the staff of automobile recall management according to the typical complaint information in a period of time. There are four levels, "high" means higher recall risk, "medium" means general recall risk, "low" means lower recall risk, and "none" means almost no recall risk. In-formation conferences are held quarterly to discuss the risk levels of some typical defect cases and to take different recall management measures for cases with different risk levels. This paper selects the results of the first information consultation as the correct risk level of the case data. In the following comparative experiments, the results of the latest meeting were selected to do the corresponding comparison and analysis.

In order to facilitate the subsequent processing of automobile defect information, it is necessary to preprocess the data in the data set. First remove duplicate and similar defect information, change the null and missing values to default values, and then normalize the car brand, manufacturer, and model. We find that there are some ambiguities in the fault labels, the fault labels are standardized in the defect information according to the classification system of the defect management center.

Automobile defective product risk recall prediction is actually a multi-classification problem. It can be found from the table that the amount of data of different risk levels is quite different, therefore, it is very important to solve the imbalance problem of data category during multi-classifying.

The purpose of data analysis is to extract the key information which may reflect the risk of automobile recall from the above defect information. After many discussions with experts of automobile recall research in the Defect Management Center, we summarized three kinds of characteristics: static risk characteristics, dynamic semantic characteristics and fault semantic characteristics. Static characteristics mainly include the brand, model, manufacturer, and defect information types. Dynamic features include mileage and years of car purchase. Fault semantic features include fault labels and fault severity levels. These features can describe and reflect the risk information hidden in defect information from different dimensions. The static features can be obtained directly from the dataset, while the dynamic features can be obtained from the defect information of the latest time stamp. Fault label features are selected from the assembly and fault label through natural language processing techniques. These features will be used as the input of the model, and provide a comprehensive and rich feature basis for the automobile defective product recall risk prediction model.

Through the statistics and analysis of the data set of automobile defect cases, we find that the automobile fault description and the automobile fault label also have certain influence on the recall risk level. The SVM model mentioned above only deals with numeric features, which is unable to capture semantic information in the fault description. The semantic representation of the text directly determines the accuracy of vehicle recall risk prediction. Bert language pre-training model is firstly used in this paper to predict the risk of automobile recall.

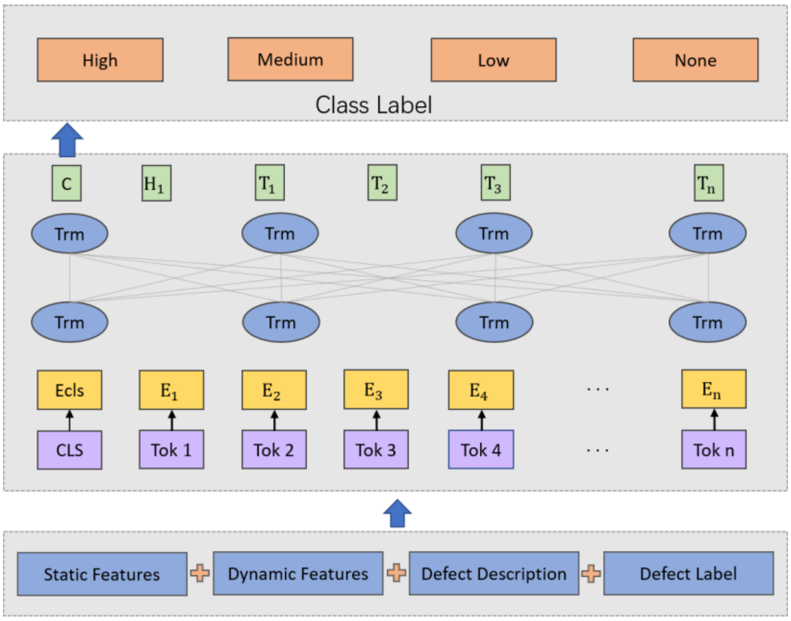


**Fig. 3.** The Architecture of VDRF based on Bert Model

The prediction task of automobile recall risk level in this paper can be regarded as a basic text classification task. Therefore, the modification of the network structure is very simple, only the first output of the last layer of Transformer needs to be used as the sentence label.

The model structure diagram of the modified Bert model used in VDRF is shown in Figure 3.

The core idea of attention mechanism used by Transformer is to calculate the relationship between each word in a sentence and all the words in the sentence, and then to think that the relationship between these words reflects the relevance and importance of different words in the sentence to some extent. Therefore, by using these relationships to adjust the importance (weight) of each word, a new expression of each word can be obtained. This new representation not only contains the word itself, but also contains the relationship between other words and the word, so it is a more global expression than a simple word vector. Transformer obtains the final text representation by continually overlapping the input text with this attention mechanism layer and the normal non-linear layer.

Therefore, the prediction of recall risk is essentially a supervised multi-classification problem, in order to accurately predict the risk level from the multi-dimensional heterogeneous defect information characteristics. In this paper, we use SVM and Bert models to predict the risk of different feature combinations of defect cases based on the existing machine learning and deep learning technologies.

Because SVM is suitable for dealing with discrete data, it is necessary to deal with the static, dynamic and fault characteristics first. For numeric class features, they are entered directly into SVM, and for class features, the one-hot method is used to convert them to 0-1 vectors. For models, assembly information, and fault labels, an index dictionary is built to numeralize the features. And then normalize the features. By selecting the appropriate kernel function and decision function, a text classifier can be obtained.

Because SVM ignores the semantic information in the fault features, natural language processing technology is used to obtain the semantic information in this paper, which may reflect the recall risk of the vehicle from another perspective.

Static features, dynamic features and fault features are used as sequence input data. Then, based on the pre-trained Chinese Bert model, the last layer of the network is reconstructed, and the text classification task based on Bert is done.

## 4. Experimental Results and Analysis

In this section, we evaluate method of the multi-label classification of vehicle complaint data and the method of the risk level prediction in the corresponding corpus. The corpus used in the experiments will be described firstly. Then the experimental results will be analyzed and discussed in the following sections.

### 4.1. Experimental Datasets

*DPAC Corpus*. This dataset is provided by the defect information collection system of Defective Product Administrative Center. It contains more than 130,000 pieces of vehicle defect complaint information, which contain one or more defect labels marked by experts in 22,747 pieces of data. These defect labels are from the Vehicle Defect Label Library of the Defective Product Administrative Center, which contains 934 defect labels. The number of defect labels and the samples of data are listed in Table 3.

**Table 3.** DPAC corpus Statistical tables

| The number of label | 1 | 2 | 3 | >=4 |
|---|---|---|---|---|
| 22747 | 16351 | 4991 | 1183 | 222 |
| Percentage | 71% | 23% | 5% | 1% |

*AUTO Corpus.* It is a new large dataset form a vehicle complain website by our crawler system. It contains more than 200,000 descriptions of complaints about defects in vehicles. All of the defect information is labeled by experts. These defect labels come from the vehicle defect classification label library of the vehicle complain website, with a total of 402 defect labels. The number of defect labels and the samples of data are listed in Table 4.

**Table 4.** AUTO corpus Statistical tables

| The number of label | 1 | 2 | 3 | >=4 |
|---|---|---|---|---|
| 200000 | 136701 | 44814 | 12871 | 5601 |
| Percentage | 68% | 22% | 6% | 4% |

*DCRL Corpus.* The automobile defect case risk level data set contains 10,351 typical automobile defect cases. The statistics of different risk levels are shown in Table 5 below.

**Table 5.** DCRL corpus Statistical tables

| Risk level | High | Medium | Low | None | Total |
|---|---|---|---|---|---|
| Number | 854 | 704 | 2944 | 5849 | 10351 |
| Percentage | 8.4% | 6.7% | 28.4% | 56.5% | 100% |

### 4.2. Evaluation Metrics

Hamming-loss [27], Micro-F1 [28] and Macro-averaging are used indicators in multi-label classification tasks [29].

### 4.3. Experimental Details

Our experiments have two main parts. The first is multi-label classification experiments. And the other is risk prediction experiments. For multi-label classification experiments, the most representative multi-label classification algorithms are selected as baseline, and the comparative experiments are carried out in large-scale corpora (AUTO corpus) and small-scale corpora (DPAC corpus).

In Multi label classification experiment, the pre-trained vehicle domain word vector model is used as word representation. In order to avoid the impact of the vehicle brand on the prediction result, synonymous substitution of the description of the vehicle brand and the vehicle system is used, and the corresponding substitution of the figures in the complaint text are also used. After statistical analysis, the first 600 words of the complaint text are intercepted as input, and the part exceeding the length of the complaint text will be discarded. Referring to the conclusion of paper [14], the frequency of the defect labels corresponding to the complaint text in the training data is sorted. The hidden state vector of the encoder and decoder is set to 300 and 600 respectively, and the number of LSTM layers of the encoder and decoder is set to 2. In the training phase, the loss function is the cross-entropy loss function. Adam optimizer is used to minimize the cross-entropy loss function [15].

That is the detailed information set during the experimenting is shown as in Table 6

**Table 6.** Parameters setting in the experiments

| Parameters | Value |
|---|---|
| Word embedding dimension | 200 |
| Label feature dimension | 32 |
| Length of beam search | 5 |
| Number of hidden layers in encoder | 300 |
| Number of hidden layers in de coder | 600 |
| Learning rate | 0.001 |
| Dropout of Learning rate | 0.5 |
| Optimizer | Adam |
| epoches | 2000 |

For the experiment of risk prediction, two methods are employed, one is based on SVM model, the other is based on Bert model. The automobile defect case data set are divided into two groups. For each risk category, 80% are selected as the training data and 20% as the test data. At the same time, different features were selected to carry out multiple sets of contrast experiments to predict the risk level of defective vehicle recall. In this experiment, there are three kinds of features, which are static features, dynamic features and fault features obtained from the automobile defect label classification experiment. Combination of these three characteristics is used as the input of the recall risk prediction model, and comparative experiments are carried out.

## 4.4. Experimental Results and Analysis

### Multi Label Classification Model

In order to evaluate the performance of different multi-label classification methods, the following five representative methods are implemented on the two dataset.

Binary Relevance (BR) [3]: transforms each label in multiple labels into a single label classification problem.

Classifier Chains (CC) [5]: transforms the multi-label classification problem into a single label classification problem, which introduces the relational information between labels in a chain structure of one label.

Label Powerset (LP) [6]: treats every possible label set combination as a new label, transforming the problem into a multi-classification problem with a single label.

CNN-RNN [12]: Global and local text semantics and label dependencies are captured using CNN and RNN, and label sequences are predicted using RNN.

The Sequence Generation Model (SGM) [14]: transforms the multi-label classification problem into a sequence generation problem, and generates a label sequence using a global-embedding decoder architecture.

We implement the BR and CC algorithms using the open source multi-label classification toolkit Scikit-Multilearn [31], and use Support Vector Machine (SVM) as the basic classifier in these algorithms [32][33].

**Table 7.** Label prediction results comparison

| Corpus | AUTO | | DPAC | |
|---|---|---|---|---|
| Metrics | Hamming Loss | Micro-F1 | Hamming Loss | Micro-F1 |
| BR-BF | 0.0106 | 0.5996 | 0.0529 | 0.5517 |
| BR-W2V | 0.0038 | 0.6301 | 0.0319 | 0.6103 |
| CC-BF | 0.0087 | 0.6176 | 0.0473 | 0.5885 |
| CC- W2V | 0.0031 | 0.6565 | 0.0297 | 0.6237 |
| LP-BF | 0.0097 | 0.6028 | 0.0476 | 0.5904 |
| LP-W2V | 0.0032 | 0.6468 | 0.0415 | 0.6175 |
| CNN-RNN | 0.0031 | 0.6971 | 0.0178 | 0.6412 |
| SGM | 0.0027 | 0.7203 | 0.0125 | 0.6563 |
| Seq2seq | 0.0028 | 0.7195 | 0.0129 | 0.6511 |

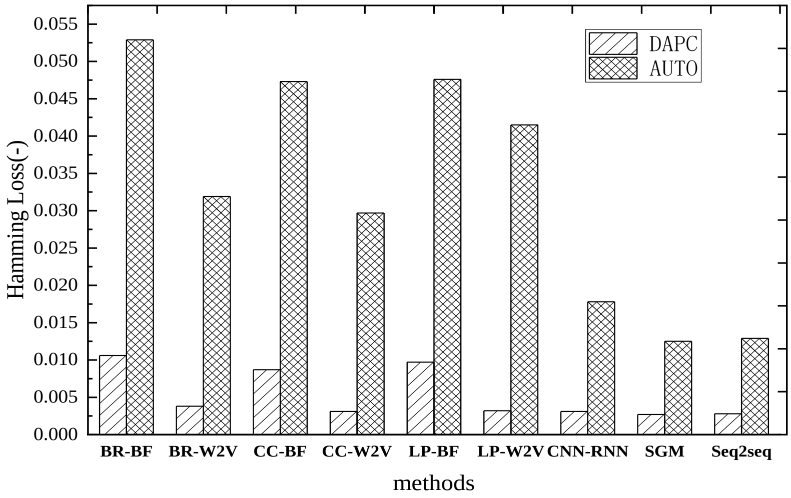


**Fig. 4.** Comparison of Hamming Loss

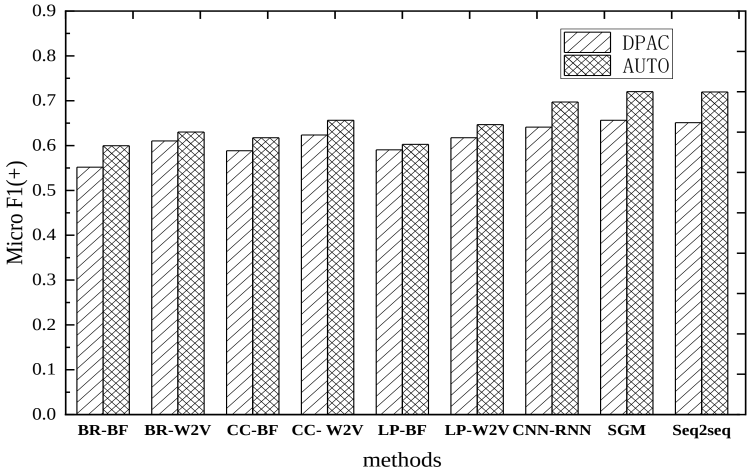


**Fig. 5.** Comparison of Micro-F1

Based on pre-trained vehicle domain word vectors, five typical multi-label classification methods are tested on two vehicle complaint datasets. The experimental results are shown in the following Table 7, Figure 4 and Figure 5, where BR stands for Binary Relevance algorithm, CC stands for Classifier Chains algorithm, BF stands for feature extraction based on vehicle defect labels, and LE stands for adding defect labels distribution vectors at the decoding layer.

In BR, CC, and LP algorithms, for a complaint text containing m words, the pre-trained domain word vector model is used to obtain the word representation vector of each word, and then the average value is obtained to represent the complaint text.

The following conclusions can be drawn from the above experiment results:

(1) Neural network-based methods are better than those using traditional multi-label classification, which shows that the neural network can recognize text information better and improve the accuracy of classification in multi-label classification.

(2) In the traditional machine learning multi-label classification method, the selection of text features has a great influence on the prediction results. From the table, it can be seen that for the same method, the result of using pre-trained domain word vectors is better than that of using label-only database features to express the complaint text, which verifies the necessity of pre-trained domain word vector model.

(3) Compared with the BR algorithm and the CC algorithm, the Classifier Chains algorithm performs better because the multiple defect descriptions contained in the vehicle complaint data are generally related to each other, and the CC algorithm takes into account the relationship between the labels. Because LP algorithm transforms the problem of multi-label classification into the problem of multi-class classification in single-label learning, and there are many kinds of multi-label combinations in the data analysis and statistics, LP algorithm is not suitable to solve this problem, and the experimental results also prove this point.

(4) Compared with CNN-RNN model, seq2seq model performs better in multi-classification of Chinese complaint texts. The reason is that seq2seq model reads the semantic information before and after each word in the complaint texts through Bi-LSTM, and pays attention to the words related to the predicted failure results through attention mechanism. CNN-RNN focuses on the high-order relevance of labels, but the recognition of the semantic information of the text itself is insufficient.

(5) Comparing SGM model with seq2seq model with attention mechanism, the input of SGM model and seq2seq model is based on pre-trained vehicle domain word vector model, and the value of word vector is allowed to change during the training process, because SGM model is based on seq2seq model with mask module and global embedded information (global embedded) in the decoder part. Experiments show that the mask module and global embedding vector are equally effective in vehicle complaint dataset. In analyzing the classification results of seq2seq model, we also find that the prediction results of the same article text contain some duplicate labels.

Based on the above conclusions, we add the feature of extended vehicle defect label library (CF) to the input layer of seq2seq model with attention mechanism. Considering the diversity of vehicle defect label combinations, a label distribution vector (LE) of each vector is obtained by using the training method of word2vec based on the defect label text of all data. A comparative experiment was carried out in two datasets. The results are shown in Table 8, Figure 6 and Figure 7.

**Table 8.** Label prediction results comparison

| Corpus | AUTO | | DPAC | |
|---|---|---|---|---|
| Metrics | Hamming Loss | Micro-F1 | Hamming Loss | Micro-F1 |
| Seq2seq | 0.0028 | 0.7195 | 0.0129 | 0.6511 |
| SGM | 0.0027 | 0.7203 | 0.0125 | 0.6563 |
| Seq2seq+CF | 0.0026 | 0.7212 | 0.0121 | 0.6532 |
| Seq2seq+CF+LE (VDIF-M) | 0.0025 | 0.7363 | 0.0100 | 0.6624 |

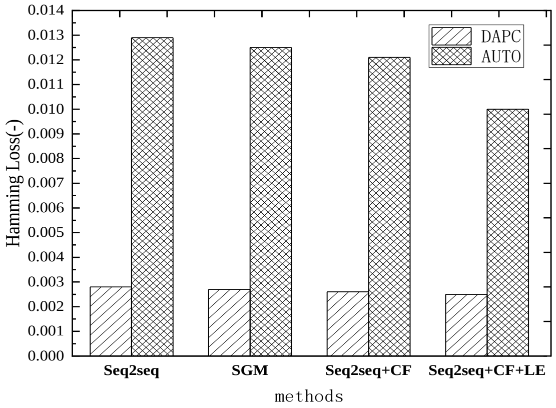


**Fig. 6.** Comparison of Hamming Loss

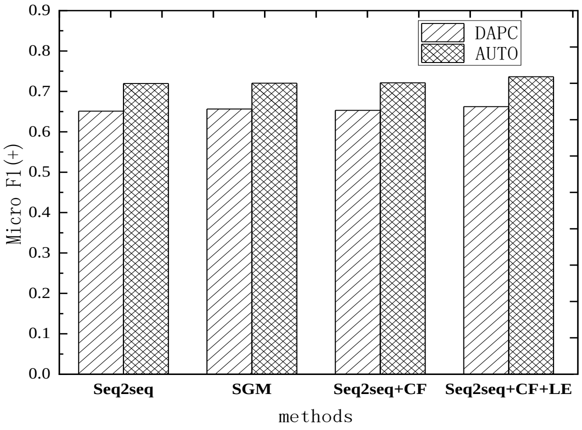


**Fig. 7.** Comparison of Micro-F1

The experimental results in the table show that the label library features added have obvious effect on the auto dataset, and the reason may be that there are fewer defect categories in the vehicle quality network, but there are more defect labels in the dataset of DPAC corpus, so the effect of adding label library features is not obvious. After the label distribution vector is added to the decoder layer, it is improved both in two datasets. Comparing with the SGM model, the experimental results show that the proposed method is superior to the SGM model in two datasets, because our methods adds defect label features suitable for vehicle complaint data, and uses the pre-trained domain word vector model at the same time.

Table 9 shows some instances of a multi-label classification that uses the different sequence models to identify only the “Engine Abnormal Noise” label in the defect description. Our proposed VDIF-M model can not only recognize the "engine-abnormal noise" label, but also generate the "Body Vibration" label ac-cording to those words "vehicle" and "jitter". This is because the extended fault description synonymous label library contains synonymous relationships between "vehicle resonance" and "vehicle jitter", which verify the model proposed in this paper can solve the multi-label classification problem of some instances by adding defect label features.

**Table 9.** DCRL corpus Statistical tables

| Defect description | VDIF-M | Seq2seq | Correct Label |
|---|---|---|---|
| 发动机有明显异响，我不懂车都能听出来，而且车辆抖动，去店里检查，说什么都正常，抖动也正常。 | 发动机-异响<br>车身附件及电器-车身共振 | 发动机-异响 | 发动机-异响<br>车身附件及电器-车身共振 |
| The engine is obviously abnormal, | Abnormal | Abnormal | Abnormal engine |

| | | | |
|---|---|---|---|
| don't understand the car can hear, and the car jitter, go to the store to check, say what is normal, jitter is normal. | engine noise<br>Body Vibration | engine noise | noise<br>Body Vibration |
| 挂d挡速度上升到40时发动机转速达到4000，但车速不上升；挂 r 挡后退无力踩住刹车时，车身抖动严重．去4s店检测,说是变速箱的3-5模块损坏,要大修变速箱。 | 发动机-无法提速<br>变速器-电脑板故障 | 发动机-无法提速<br>变速器-异响 | 发动机-无法提速<br>变速器-电脑板故障 |
| When the speed of the gearbox increases to 40, the speed of the engine reaches 4000, but the speed of the car does not rise; when the gearbox is unable to step on the brake, the body shakes seriously. Go to 4S shop to check that the 3-5 module of the gearbox is damaged, it is necessary to overhaul the gearbox. | Engine Unable to Speed up<br>Transmission-Computer Board Failure | Engine Unable to Speed up<br>Transmission Abnormal engine noise | Engine Unable to Speed up<br>Transmission Computer Board Failure |

### Recall Risk Prediction Model

In order to accurately predict the risk level from the multi-dimensional heterogeneous defect information features, SVM and Bert models are used to predict the risk of different defect case feature combinations based on the existing machine learning and deep learning technologies

In the SVM experiment, one-hot to represent the class information in the static feature directly. For the automobile brand, manufacturer and other information, an index table is built to convert the corresponding features into numerical values. The mileage in the dynamic features is in the unit of 10,000 km and the service life is in the unit of years. In this experiment, we first use different feature combinations, choose the kernel function as Gaussian kernel, and the penalty coefficient is 1, the class weight is the default.

From Table 10, it can be seen that different combinations of features have different effects on recall risk. Static features have the greatest impact on recall risk, dynamic features have the least impact, and the effect of three types of features fusion is the best.

**Table 10.** Different Feature Combination Result-SVM

| Feature combination | Parameters | Macro-acc | Macro-recall | Macro-f1 |
|---|---|---|---|---|
| $\vec{X}s$ | default | 0.62 | 0.55 | 0.57 |
| $\vec{X}d$ | default | 0.55 | 0.56 | 0.55 |
| $\vec{X}l$ | default | 0.60 | 0.54 | 0.56 |
| $\vec{X}s\,\vec{X}d$ | default | 0.62 | 0.67 | 0.64 |
| $\vec{X}s\,\vec{X}l$ | default | 0.65 | 0.69 | 0.66 |
| $\vec{X}d\,\vec{X}l$ | default | 0.61 | 0.55 | 0.57 |
| $\vec{X}s\,\vec{X}d\,\vec{X}l$ | default | 0.80 | 0.68 | 0.72 |

Although the overall results of the experiment are very good, for the categories with a small number of samples, the prediction ability of SVM is very limited.

In the experiment of Bert model [34], we forecast the recall risk of defect cases based on the pre-trained Chinese language model Bert model published by Google. Because Bert model is more suitable for processing sequence data, dynamic features, static features, defect features and the combination of the three features are used for comparative experiments. At the same time, we adjust the learning rate and the maximum sequence length in the training process to get the most suitable combination of parameters for this task. The achieved experimental results are shown in Table 11.

**Table 11.** Different Feature Combination Result-Bert

| Feature combination | Macro-acc | Macro-recall | Macro-f1 |
|---|---|---|---|
| $\vec{X}s$ | 0.65 | 0.59 | 0.61 |
| $\vec{X}d$ | 0.59 | 0.56 | 0.57 |
| $\vec{X}l$ | 0.61 | 0.59 | 0.6 |
| $\vec{X}s\,\vec{X}d$ | 0.71 | 0.65 | 0.67 |
| $\vec{X}s\,\vec{X}l$ | 0.73 | 0.62 | 0.67 |
| $\vec{X}d\,\vec{X}l$ | 0.64 | 0.60 | 0.62 |
| $\vec{X}s\,\vec{X}d\,\vec{X}l$ | 0.79 | 0.78 | 0.79 |

From Table 11, it can be found that the combination of the three features has achieved best results in the process of risk prediction, which is the most suitable parameter combination for this task.

Detailed metrics comparison between the Bert and SVM models are listed in Table 12. As can be seen from the above table, the prediction effect of Bert and SVM is not much different for the categories with more sample data. Bert model is more accurate in predicting the smaller sample categories, which also shows that Bert model can solve the sample imbalance problem to some extent.

**Table 12.** Detailed comparison between SVM and Bert

| Model | Risk Level | Accuracy | Recall rate | F1-score | Number of samples |
|---|---|---|---|---|---|
| Bert | high | 0.89 | 0.82 | 0.85 | 171 |
| | medium | 0.59 | 0.62 | 0.60 | 141 |

| | | | | | |
|---|---|---|---|---|---|
| | low | 0.80 | 0.80 | 0.80 | 589 |
| | none | 0.88 | 0.89 | 0.89 | 1170 |
| | micro-avg | 0.84 | 0.84 | 0.84 | 2071 |
| | macro-avg | 0.79 | 0.78 | 0.79 | 2071 |
| | weighted-avg | 0.84 | 0.84 | 0.84 | 2071 |
| SVM | high | 0.89 | 0.74 | 0.81 | 171 |
| | medium | 0.43 | 0.62 | 0.51 | 141 |
| | low | 0.90 | 0.84 | 0.87 | 589 |
| | none | 0.76 | 0.83 | 0.79 | 1170 |
| | micro-avg | 0.81 | 0.81 | 0.81 | 2071 |
| | macro-avg | 0.75 | 0.75 | 0.74 | 2071 |
| | weighted-avg | 0.83 | 0.81 | 0.82 | 2071 |

In most multi-classification tasks, the category with smaller sample size is usually the most concerned. In this task, "medium" risk and "high" risk category have the smaller sample size. Therefore, they are the most important supervision objects of defect recall management. In cases where the recall risk is neutral and high, appropriate measures are usually taken in the subsequent process. High-risk and medium-risk samples are similar, and the characteristics are not clear in the process of multi-classification, which is also one of the reasons for the low accuracy of high-risk and medium-risk cases in this topic.

## 5. Conclusion and Future Work

In the management of automobile recall, the risk assessment of automobile defect cases is the basis of the follow-up supervision. In order to improve the efficiency of defect risk rating assessment, this paper presents a model for predicting the risk level of automobile recall based on defect information. After building the defect case dataset, the multi-dimensional features of case data are extracted by data analysis, Bert model is used to read the fault information from the defect information, and different feature combinations and different models are compared with each other. The experiments show that the recall risk prediction model based on Bert model has best performace both in classification and prediction tasks. which can provide a powerful reference for automobile defect experts.

In the research of automobile recall risk prediction, only the defect information of automobile recall cases is collected by the defective product management center, most of which are the inherent attribute values and dynamic attribute characteristics of automobile. Although it also contains a public opinion index feature, the public opinion features of each case are not different. In the follow-up risk prediction work, mining the relevant industry news, complaint news, user comments and other information of different automobile brands is the main work in the future of automobile recall risk level prediction.

**Acknowledgements.** This work is supported by National Natural Science Foundation of China under Grants No. 61671070, 61972364. Defense-related Science and Technology Key Lab Fund project 6142006190301, National Language Committee of China under Grants ZDI135-53, and Project of Developing University Intension for Improving the Level of Scientific Research–No.2019KYNH226, Qin Xin Talents Cultivation Program, Beijing Information Science & Technology University No.QXTCP B201908.

## References

1. Defective Product Administrative Center Homepage, http://www.dpac.gov.cn, last accessed 2019/1/24.
2. Boutell, M. R., Luo, J., Shen, X., Brown, C. M.:Learning multi-label scene classification. Pattern recognition.37, 1757-1771.( 2004)
3. Zhang, M. L., Wu, L.:Lift: Multi-label learning with label-specific features. IEEE transactions on pattern analysis and machine intelligence, 37, 107-120.(2014)
4. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining (IJDWM), 3, 1-13.(2007)
5. Read, J., Pfahringer, B., Holmes, G., Frank, E.:Classifier chains for multi-label classification. Machine learning, 85, 333.( 2011)
6. Clare, A., King, R. D.:Knowledge discovery in multi-label phenotype data. In European Conference on Principles of Data Mining and Knowledge Discovery. Springer, Berlin, Heidelberg,42-53.(2001)
7. Gao, H.H., Liu,C., Li, Y.H.Z., Yang, X.X. V2VR: Reliable Hybrid-Network-Oriented V2V Data Transmission and Routing Considering RSUs and Connectivity Probability[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, DOI: 10.1109/TITS.2020.2983835
8. Elisseeff, A., Weston, J.:A kernel method for multi-labelled classification. In Advances in neural information processing systems.681-687.(2002)
9. Zhang, M.L., Zhou, Z.H.:ML-KNN: A lazy learning approach to multi-label learning [J]. Pattern Recognition, 40,2038-2048.( 2007)
10. Li, L., Wang, H., Sun, X.:Multi-label Text Categorization with Joint Learning Predictions-as-Features Method[C]// Conference on Empirical Methods in Natural Language Processing.835-839.(2015)
11. Zhang, M. L., Zhou, Z. H.:Multi-label Neural Networks with Applications to Functional Genomics and Text Categorization [J]. IEEE Transactions on Knowledge & Data Engineering, 18,1338-1351.(2006)
12. Chen, G., Ye, D., Xing, Z.:Ensemble application of convolutional and recurrent neural networks for multilabel text categorization[C]// International Joint Conference on Neural Networks. IEEE, 2377-2383.( 2017)
13. Baker, S., Korhonen, A.:Initializing neural networks for hierarchical multi-label text classification. InBioNLP.307-315.(2017)
14. Yang, P., Sun, X., Li, W.:SGM: Sequence Generation Model for Multi-label Classification [J].(2018)
15. Gao, H.H., Kuang, L., Yin, Y.Y., Bin Guo, B.,Dou, K.Mining Consuming Behaviors with Temporal Evolution for Personalized Recommendation in Mobile Marketing Apps [J]. ACM/Springer Mobile Networks and Applications (MONET), 2020, DOI: 10.1007/s11036-020-01535-1
16. Zhang, X., Niu, S., Zhang, D.; Wang, G. A.; Fan, W.:Predicting vehicle recalls with user-generated contents: A text mining approach. In Pacific-Asia Workshop on Intelligence and Security Informatics. Springer, Cham.41-50.( 2015)
17. Yang, S.L. :Research on Early Warning Model of Defective Automobile Recall Risk Based on Internet Complaint Data [D].Beijing Information Science and Technology School.(2015)

18. Jiang, C. Q, Wang, Q.L., Liu, S.X., Ding, Y., Liu, R. :Semi supervised learning method for vehicle defect recognition in Chinese social media environment [J]. Chinese Journal of Management Science, 22,677-685.( 2014)
19. Abrahams, A. S., Jiao, J., Wang, G. A., Fan, W.:Vehicle defect discovery from social media. Decision Support Systems.54, 87-97.( 2012)
20. Abrahams, S., Jiao, J., Fan, W., Wang, G. A., Zhang, Z. :What's buzzing in the blizzard of buzz. Automotive component isolation in social media postings Decis. Support Syst, 55.( 2013)
21. Gao, H.H., Xu, Y.S.,Yin, Y.Y., Zhang, W.P., et al. Context-aware QoS Prediction with Neural Collaborative Filtering for Internet-of-Things Services [J]. IEEE Internet of Things Journal(IoT-J) , 7(5): 4532-4542 (2020)
22. Chinese synonyms Toolkit. https://github.com/huyingxi/Synonyms, last accessed 2019/1/24.
23. Chinese Word Segmentation Tool. https://pypi.org/project/jieba/, last accessed 2019/1/24.
24. Zhang, Y.W., Li, B.A., Lv, X.Q. :Research on Domain Term Dictionary Construction based on Chinese Wikipedia, Image Processing, Computing and Big Data .(2018)
25. Mikolov, T., Chen, K., Corrado, G., Dean, J. :Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.(2013)
26. Graves, A., Schmidhuber, J.:Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural networks, 18, 602-610.( 2005)
27. Schapire, R. E., Singer, Y.:Improved boosting algorithms using confidence-rated predictions. Machine learning, 37, 297-336.(1999)
28. Manning, C. D., PRABHAKAR, R., HINRICH, S.:Introduction to information retrieval, volume 1 Cambridge University Press. Cambridge, UK.(2008)
29. Ma, X.J., Gao, H.H., Xu, H.H., Bian, M.J.. An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing. EURASIP Journal on Wireless Communications and Networking [J], 2019, 2019(249).(2019)
30. Wiseman, S., Rush, A. M.:Sequence-to-sequence learning as beam-search optimization. arXiv preprint arXiv:1606.02960.(2016)
31. Szymański, P., Kajdanowicz, T. A. :scikit-based Python environment for performing multi-label classification. arXiv preprint arXiv:1702.01460.(2017)
32. Yang, X.X., Xiaoxian, Zhou,S.J., Cao, M. . An Approach to Alleviate the Sparsity Problem of Hybrid Collaborative Filtering Based Recommendations: The Product-Attribute Perspective from User Reviews. MOBILE NETWORKS & APPLICATIONS, 25(2):376-390 (2020)
33. Zhu, R., Yu, D., Ji, S., Lu, M. :Matching RGB and Infrared Remote Sensing Images with Densely-Connected Convolutional Neural Networks. Remote Sens.11, 2836.(2019)
34. Lin, C., Miller, T., Dligach, D., Bethard, S., Savova, G. :A BERT-based universal model for both within-and cross-sentence clinical temporal relation extraction. In Proceedings of the 2nd Clinical Natural Language Processing Workshop.65-71.(2019)

**Xindong You** is currently an associate professor of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research at Beijing Information Science & Technology University, China. She holds a post-doctoral position at Tsinghua University with the Beijing Institute of Graphic Communication, from 2016 to 2018. Before as a post-doctoral, she is an Associate Professor in Hangzhou Dianzi University. Before joining Hangzhou Dianzi University, she was a Ph.D. student with Northeastern University, from 2002 to 2007. She received her PhD degree in 2007. She is in charge of the National Nature Science Foundation of China from 2014 to 2017 and Nature Science Funding of Zhejiang Province from 2013 to 2015. She has authored about 30 papers in the international conference or journals, most of them are indexed by EI or SCI database. Her current research areas include Natural Language Processing, Image Processing, Distributed Computing, Cloud Storage, Energy Management, Data Replica Management, etc.

**Jiangwei Ma** is currently a postgraduate student in Beijing Information Science & Technology University, China. She received her B.S. degree from Beijing Information Science & Technology University, China, in 2018. Her current research areas include Relation Extraction, Natural Language Processing, etc.

**Yuwen Zhang** was a graduate student at Beijing Information Science & Technology University, China. He received his master degree from Beijing Information Science & Technology University, China, in 2019. His current research areas include Automobile Risk Level Detection, Natural Language Processing, etc.

**Xueqiang Lv** is a professor in Beijing Information Science & Technology University. Before joining in Beijing Information & Technology University, he is a post-doctoral of Peking University from 2003 to 2005. Before as a post-doctoral, he is a PhD candidate in Northeastern University from 1998 to 2003. He received his PhD degree in 2003. Until now, he has been in charge of the National Nature Science Foundation of China three times. He has authored about 60 papers in the international conference or journals, most of them are indexed by EI or SCI database. His current research areas include Cloud Computing, Distributed Computing, Natural Language Processing, Image Processing, Information retrieval, Machine Learning, Deep Learning, etc.

**Junmei Han** is currently an associate professor of Department of Systems General Design Institute of Systems Engineering. AMS. PLA, China. She received the M.S. degree in Military Communication and PhD degree in information and communication engineering from National University of Defense Technology, Changsha, China, in 2013 and 2019, respectively. Her current research areas include Complex System Theory, Natural Language Processing, Graph neural network, etc.

*Received: September 03, 2020; Accepted: April 23, 2020*

Computer Science and Information Systems 17(3):819–834 https://doi.org/10.2298/CSIS190923022O

# A Homomorphic-encryption-based Vertical Federated Learning Scheme for Rick Management

Wei Ou[1], Jianhuan Zeng[2], Zijun Guo[1], Wanqin Yan[1], Dingwan Liu[1],
and Stelios Fuentes[3]

[1] Department of Electronic and Information Engineering
Hunan University of Science and Engineering, Yongzhou, China
{ouwei1978430, yanwanqinqin, liudinwan}@163.com, GuoZijun0831@gmail.com
[2] Artificial Intelligence Research Center
Qianhai Institute for Innovation Research, Shenzhen, China
zengjianhuan@foxmail.com
[3] Leicester University, UK
stelios.fuentes@gmx.co.uk

**Abstract.** With continuous improvements of computing power, great progresses in algorithms and massive growth of data, artificial intelligence technologies have entered the third rapid development era. However, With the great improvements in artificial intelligence and the arrival of the era of big data, contradictions between data sharing and user data privacy have become increasingly prominent. Federated learning is a technology that can ensure the user privacy and train a better model from different data providers. In this paper, we design a vertical federated learning system for the for Bayesian machine learning with the homomorphic encryption. During the training progress, raw data are leaving locally, and encrypted model information is exchanged. The model trained by this system is comparable (up to 90%) to those models trained by a single union server under the consideration of privacy. This system can be widely used in risk control, medical, financial, education and other fields. It is of great significance to solve data islands problem and protect users' privacy.

**Keywords:** Data Security, Privacy Preservation, Federated Learning, EM Algorithm, Homomorphic Encryption.

## 1. Introduction

With the arrival of the era of big data, more and more industries are paying attention to artificial intelligence technology. The ability of artificial intelligence to quickly process large amounts of data and the ability to mine hidden information links between discrete data has great advantages in the financial industry. Traditional risk management has always been based on manual experience, relying on the experience of the risk managers to make decisions. However, even the most experienced risk manager cannot make the correct or optimal decision for every potential risk every time. Machine learning can make predictions of credit risk quickly, and often more accurate than the results of manual predictions [1][2][3][4]. However, machine learning and deep learning often require a large amount of high-quality raw data for training [5][6] in order to obtain models that can effectively predict or judge.

To solve the dilemma of big data, the traditional method has become a bottleneck. Simple data exchange between two enterprises is not allowed in many regulations, including the GDPR (General Data Protection Regulation). Users are the owners of the original data. Without the approval of users, enterprises cannot exchange data. Secondly, the purpose of data modeling cannot be changed before user approval. Therefore, many attempts of data exchange in the past, such as data exchange, also need great changes to comply. At the same time, the data owned by enterprises often has great potential value. Two enterprises and even the departments within one enterprise may consider the exchange of interests. Under this premise, these departments often do not simply aggregate data with other departments. It will result in data frequently appearing as silos even within the same company.

In this case, Federated Learning incarnates its advantages. Unlike conventional machine learning, Federated Learning does not require companies to exchange data with each other to implement model training. Keeping the data locally for training means that using the data will not violate privacy protection regulations, nor will it result in the disclosure of corporate trade secrets. This is of great significance for solving the problem that SMEs do not have enough high-quality data to train models and how to protect the information security of users and enterprises.

In this paper, we combine the vertical federated learning with Bayesian Machine Learning and use the homomorphic encryption algorithm to design a federated learning system which is secure, stable, and effective. With a view to solve data silos between enterprises and protect user privacy.

## 2. Related Works

### 2.1. Federated Learning

Because of Alphago’s great success people naturally hope that the data-driven AI will be realized in all walks of life, but the real situation is very disappointing. In addition to a limited number of industries, there are more areas with limited data and poor quality, which is not enough to support the implementation of artificial intelligence technology. There are two main reasons: 1) There are barriers between data sources that are hard to break. In most industries, data exists in the form of silos. Due to industry competition, user privacy, complicated administrative procedures and other issues, even data integration between different departments of the same company faces many obstacles. In reality, it is almost impossible to integrate the scattered data in different places and institutions. In other words, the cost is huge. 2) With the further development of big data, it has become a worldwide trend to attach importance to user privacy and data security. Every time the public data is leaked, it will cause great concern of the media and the public. At present, all countries are strengthening the protection of data security and user privacy. The increasingly strict management of user privacy and data security will be the worldwide trend, which brings unprecedented challenges to the field of artificial intelligence. The current situation in the research and business is that the party who collects data is usually not the party who uses the data. For example, Party A collects data, transfers it to Party B for cleaning, then transfers it to Party C for modeling, and finally sells this model to Party D for use. This form of data transfer, exchange and transaction between entities violates the GDPR and may be severely punished by the laws. Similarly, *Cybersecurity Law of the People's Republic of China* and *General Provisions of the Civil Law of the People's Republic of China* which have been implemented since 2017, also point out that network operators can not disclose, tamper with or destroy the personal information they collect, and when conducting data transactions with third parties, it is necessary to ensure that the proposed contract clearly stipulates the scope and data protection obligations of the data to be traded. The establishment of these laws and regulations challenges the traditional data processing mode of AI in different degrees.

Therefore, how to design a machine learning framework on the premise of meeting the requirements of user privacy, data security and supervision, so that the AI system can use their data more efficiently and accurately, is an important topic in the development of AI.

Federated Learning is a solution proposed by many institutions and scholars to the dilemma of data isolation and data privacy. For the privacy of mobile terminal and multi-party organization data, Google and WeBank have built their different Federated Learning frameworks. Google's Federated Learning framework is based on personal terminal devices, and AAAI Fellow Prof. Yang Qiang and WeBank subsequently proposed a systematic and general solution based on Federated Learning, which can give a solution to the difficulty of co-modeling between individuals or companies. On the premise of meeting data privacy, security and regulatory requirements, a machine learning framework is designed to enable artificial intelligence systems to use their data more efficiently and accurately. Currently, the main research directions of Federated Learning are to overcome the statistical challenges and enhance security. Prof. Yang Qiang uses the gradient descent algorithm to train a linear regression model, and the model built in combination with the homomorphic encryption algorithm is a very good case. Federated Learning is in the initial stage of rapid development. Both WeBank and Google have launched their own open source Federated Learning frameworks, FATE (Federated AI Technology Enabler) and TensorFlow Federated. In order to accelerate the popularization and implementation of "Federated Learning", WeBank submitted a proposal to IEEE Standards Association in October 2018, "Guide for Architectural Framework and Application of Federated Machine Learning" (Federated Learning Infrastructure and Application Standards), which was approved in December 2018.

*(1) Basic Notion*

When considering traditional machine learning, the training process can be considered as an optimization problem, defined as:

$$\min_{\omega\in ¡^{d}} f(\omega) \tag{1}$$

$$f(\omega)\overset{\mathrm{V}}{=}\frac{1}{n}\sum_{i=1}^{n} f_i\left(\omega\right) \tag{2}$$

In above formulas, $f_i(\omega)$corresponds to the loss function; Given the parameter ω, $f_i(\omega)$ is the predicted loss at the index of data point $i$. First, the basic concept of Federated Learning is presented in combination with Figure 1. Federated learning [7] consists of two main components, including central training and local training, and the

K clients that make up the system are identified by index k. The process is divided into multiple communication rounds. In each round, clients train the local model synchronously using local SGD on their private data sets $P_k$. On the central server side, server aggregates the uploaded client parameters. Specifically, the parameter from client k is $\omega^k$, and $k \in S$. S corresponds to (in each communication round) a participating subset which contains $m$ clients.

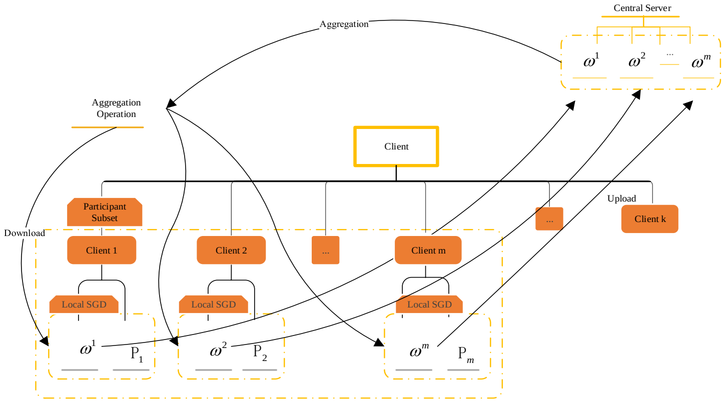


**Fig. 1.** Federal Learning System

For client k, the training data set owned by client k has $n_k$ data points, and $n_k = |P_k|$. Therefore, the optimization problem under the Federated Learning can be redefined as:

$$f(\omega)=\sum_{k=1}^{k}\frac{n_k}{n}\cdot F_k(\omega) \tag{3}$$

$$F_k(\omega)=\frac{1}{n_k}\sum_{i\in \mathrm{P}_k} f_i(\omega) \tag{4}$$

What needs to be added is that the classic assumption about training data in conventional distributed optimization, Independent and Identically Distributed Assumption (IID Assumption), training data is evenly and randomly distributed on each client, which is difficult to satisfy in the Federated Learning [8][9].

*(2) Classification of Federated Learning*

Matrix D$i$ denotes the data held by each data owner $i$. Each row of the matrix represents a user sample, and each column represents a user's feature. At the same time, some data sets may also contain label data. We called the features space as X and the label space as Y. For example, in the financial field, the user's information is the label Y that needs to be predicted; in the sales field, the label is the user's purchase wish Y; in the education field, it is the degree of knowledge of the student. User feature X and label Y constitute the complete training data $(X, Y)$. However, in implementation, it is often encountered that the users of different data sets or the user characteristics are not exactly the same. Specifically, taking Federated Learning with two data owners as an example, the data distribution can be divided into three cases [10][11][12]: 1) The features space $(X_1, X_2, \cdots)$ of the two data sets have a large overlap and the user space $(U_1, U_2, \cdots)$ have a small overlap. 2) The user space $(U_1, U_2, \cdots)$ overlap is larger and the feature space $(X_1, X_2, \cdots)$ overlap is smaller. 3) The user space $(U_1, U_2, \cdots)$ and feature space $(X_1, X_2, \cdots)$ overlaps in both datasets are small Federated Learning can be divided into Horizontal Federated Learning, Vertical Federated Learning, and Federated Transfer Learning, as shown in Fig. 2.

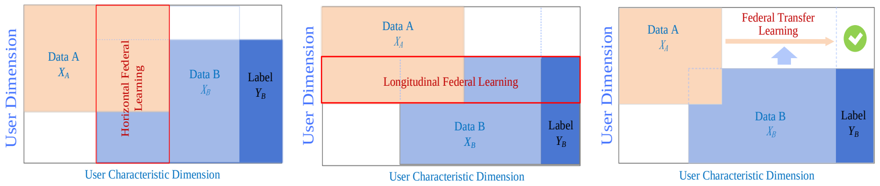

User Dimension
Data A
$X_A$
Horizontal Federal Learning
Data B
$X_B$
Label
$Y_B$
User Characteristic Dimension
User Dimension
Data A
$X_A$
Longitudinal Federal Learning
Data B
$X_B$
Label
$Y_B$
User Characteristic Dimension
User Dimension
Data A
$X_A$
Federal Transfer Learning
Data B
$X_B$
Label
$Y_B$
User Characteristic Dimension

**Fig. 2.** Classification of Federal Learning

1) Horizontal Federated Learning

In the scenarios that data sets share the same feature space but different in users, we divide the data horizontally (the user dimension), and take the part of the data with the same feature space but not the same users for training. This approach is called Horizontal Federated Learning. For example, there are two banks in different regions. Their user groups come from their respective regions, and the intersection between them is small. However, their businesses are similar, so the features recorded are the same. And then, they can use Horizontal Federated Learning to build a model together. In 2017, Google proposed a data co-modeling solution for Android phone model updates: a Federated Learning scheme that when a single user uses an Android phone, the model parameters are continuously updated locally and uploaded to the Android cloud, so that each data owner with the same feature dimension can build a model jointly.

2) Vertical Federated Learning

In the scenarios that data sets share the same users but different in feature space, we divide the data vertically (the feature space dimension), and take the part of the data with the same users but not the same feature space for training. This method is called Vertical Federated Learning. For example, there are two different institutions, one is a bank and the other is an e-commerce company in the same place. Their users are likely to include most of the residents of the place, so the intersection of users is large. However, because banks record users' payment behaviors and credit ratings, while e-commerce companies store users' browsing and purchasing history, their feature space overlap is small. Vertical Federated Learning is a scheme that aggregates these different features in an encrypted state to enhance the ability of the model. At present, many machine learning models such as logistic regression models, tree structure models, and neural networks models have gradually been proven to be able to be built on this federated system.

3) Federated Transfer Learning

In the scenarios that the user and feature space of the two data sets have little overlap, we do not split the data, but use transfer learning instead to overcome the lack of data or labels. This method is called Federated Transfer Learning. For example, there are two different institutions, one is a bank located in China, and the other is an e-commerce company located in the United States. Due to geographical constraints, the user groups of the two institutions have very little intersection. At the same time, due to types of the institutions are different, only a small part of the features overlap. In this case, in order to make federated learning system runs effectively, Federated Transfer Learning must be introduced to solve the problem of small unilateral data and small label samples, thereby improving the effectiveness of the model.

### 2.2. Homomorphic Encryption

In Federated Learning, Homomorphic Encryption is used to encrypt parameters to protect user privacy. Unlike differential privacy, the data and model will not be transmitted. So, there is no possibility of leakage at the data level, nor violation of the stringent data protection laws such as GDPR. Homomorphic Encryption [13][14][15] is a kind of encryption method that has special nature properties, this concept was first proposed in the 1970 s by Rivest et al., compared with the general encryption algorithms, homomorphic encryption can not only realize the basic cryptographic operations, but also achieve a variety of computing functions between the ciphertext [16][17]. In other words, first calculation and then decryption are equivalent to first decryption and then calculation. Gentry based on the ideal lattice for the first time in 2009 proposed the real practical Fully Homomorphic Encryption scheme (FHE). Gentry’s scheme can suppress the growth of noise by performing a finite number of polynomial operations and using the compression and decryption circuit technology when performing homomorphic operations, so as to prevent the noise from growing too fast and exceeding a certain limit causes the plaintext value cannot be decrypted correctly. So far, after two stages of development, researchers have proposed a variety of fully homomorphic encryption schemes and corresponding optimization schemes. Most of the existing homomorphic encryption schemes can only support integer type homomorphic encryption, but do not support floating point types, so it cannot cover the practical application requirements. In addition, due to the inherent reasons such as algorithm construction and security guarantee, the efficiency of the algorithm and the storage occupancy of secret key and ciphertext are far from the standards of practical production and application. How to design and propose a new Fully Homomorphic Encryption scheme on the premise of ensuring the security of the Fully Homomorphic Encryption algorithm and overcoming the disadvantages of existing algorithms in noise control, execution efficiency, storage space, etc., will be an important direction of current research.

A solution proposed by Graepel to solve the problem of excessive noise caused by homomorphic encryption operation by mathematically expressing the prediction function of the model as a low-order polynomial. The research is about privacy protection in the training stage. This scheme effectively limits the number of homomorphic operations on encrypted data and limits homomorphic operations to addition and multiplication. Finally, it is applied to LM and FLD classifier and practical results are obtained. Later, David Wu et al. realized homomorphic encryption of large-scale data sets and high-dimensional data by using batch computing and CRT-based message encoding technology, and then performed linear regression and other statistical analysis on the encrypted data, and the results were suitable for the scenario of multi-source data. To some extent, homomorphic encryption only supports limited homomorphic operations, but low-order polynomials can satisfy this property. Therefore, Dowlin uses Chebyshev approximation theory to replace the nonlinear activation function in the neural network model with a low-order polynomial function, thus realizing CryptoNets, a neural network that can process ciphertext. And through the experiment on the real data set MNIST, the rationality of the model is explained. The research is done in the classification stage, because the neural network model has better accuracy than the linear classifier, which is a good breakthrough. Bost et al. tried to process ciphertext in hyperplane decision, Naive Bayes and decision tree classifier, which were also studied in the classification stage. Since CryptoNets proposed by Dowlin did not perform well on deep neural networks, Chabanne et al. proposed improvements on this basis. The convolutional neural networks model proposed by them was more accurate than CryptoNets, which was also the first successful attempt to combine homomorphic encryption with deep neural network. In addition, Aono used additive homomorphic encryption to propose a logistic regression model that can be calculated on ciphertext. It can be seen that data based on homomorphic encryption has been applied to many common machine learning models and achieved good accuracy. However, it is worth mentioning that due to the complexity of the calculation model, the calculation time is generally much longer than that of the plaintext processing [18].

Homomorphic encryption is an encryption method that can perform any operation which can be performed on plaintext on encrypted data without decryption [16]. In other words, homomorphic encryption satisfies a specific algebraic algorithm for plaintext that is equivalent to another (possibly different) algebraic algorithm for ciphertext.

We perform some kind of processing on the encrypted data to generate a new ciphertext. The plaintext content generated by decrypting the ciphertext, is the same as the result that decrypt corresponding plaintext after encrypting operation. This method allows the ciphertext can be manipulated even when the data is encrypted, and the result of it will be the same as we expected. The process of homomorphic encryption is shown in Fig. 3. Encrypt and Decrypt represent corresponding encryption and decryption methods; the “*” operation in the plaintext space and the “#” operation in the ciphertext space are equivalent.

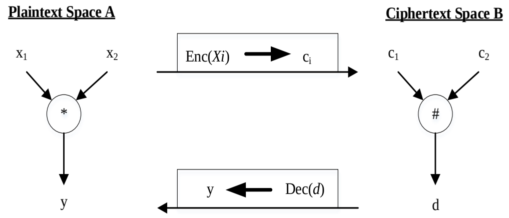


**Fig. 3.** Homomorphic Encryption Algorithm

Homomorphic encryption can only perform any number of times of additions or multiplications or finite additions and multiplications on the ciphertext, and its operation result is the same as the result of encrypting the plaintext directly after the corresponding operation.

Assume that an encryption scheme G is expressed as $(M, C, K, E, D)$, where M is the plaintext space, C is the ciphertext space, K is the key space, E is the encryption algorithm, and D is the decryption algorithm. Defines $\oplus$ as the ciphertext related operator.

**Definition 1** Let P and L denote the operation, when the plaintext data set $M = \{m_1, m_2, \cdots, m_n\}, k \in K$ , if: $P(E_k(m_1), E_K(m_2), \cdots, E_K(m_n)) = E_k(L(m_1, m_2, \cdots, m_n))$ , says that the encryption scheme for operation L is homomorphism. The basic idea of homomorphic encryption is to achieve that after doing some operations on ciphertext we can still get the same result as doing operations on plaintext directly.

**Definition 2** For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i), c_j = E(m_j)$ , and $c_i, c_j \in C$ , if $(m_i + m_j) = E(m_i) \oplus E(m_j)$ or $D\left(E(m_i) \oplus E(m_j)\right) = m_i + m_j$ holds, then it is said that the encryption scheme G has the property of addition homomorphism.

**Definition 3** For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i), c_j = E(m_j)$ , and $c_i, c_j \in C$ , if $E(m_i m_j) = E(m_i) \oplus E(m_j)$ or $D(E(m_i) \oplus E(m_j)) = m_i m_j$ holds, the encryption scheme G is said to have multiplicative homomorphism.

**Definition 4** For any plaintext $m_i, m_j \in M$, the corresponding ciphertext is $c_i = E(m_i)$, and $c_i \in C$, if $E(m_i m_j) = E(m_i) \oplus m_j$ or $D(E(m_i) \oplus m_j) = m_i m_j$ holds, then it is said that the encryption scheme G has the property of mixed multiplication homomorphism.

**Definition 5** If scheme G has both the addition homomorphism and multiplication homomorphism properties, and can satisfy finite addition and multiplication ciphertext operations, then the encryption scheme G is said to be a somewhat homomorphic encryption scheme.

**Definition 6** If scheme G has both addition homomorphism and multiplication homomorphism properties, and can satisfy any number of times of addition and multiplication ciphertext operations, then the encryption scheme G is said to be a fully homomorphic encryption scheme [19].

### 2.3. EM Algorithm

The EM algorithm [20] was formally proposed by Arthur Dempster, Nan Laird and Donald Rubin in their research paper Maximum likelihood from incomplete data via the EM algorithm in 1977. They summarized the previous EM algorithm as a special case and gave the calculation steps of the standard algorithm. After that, EM algorithm became a method to deal with incomplete observation data, which attracted much attention from all sides and was continuously studied in depth. Since 1977, there were many new applications and improvements of the algorithm [3][21]. After decades of development, the EM algorithm has been widely used to process incomplete data in medicine, engineering, business management, sociology, finance and other fields where large data volume is required. The EM algorithm is an iterative algorithm. As with most iterative algorithms, the EM algorithm requires the user to make an initial assumption about the parameters to be solved, and then continuously update the value of this set of parameters until there are no more noticeable changes. For EM algorithms, different initial parameters often lead to different results. That is, the EM algorithm cannot guarantee that the results obtained are optimal. Theoretical analysis shows that the EM algorithm can only guarantee the locally optimal solution. In addition to the local optimal problem (or the corresponding initial sensitivity) mentioned above, the EM algorithm has another tricky problem that is the user needs to pre-set the number of gaussian members in the GMM model (each gaussian function in the gaussian mixture model is called a member, or Component). This problem is tricky because the GMM model deals with a bunch of unlabeled data, meaning that it is not clear which gaussian member is responsible for each data point. How to overcome the local extremum problem (or initial sensitivity problem) of EM algorithm and how to determine the number of gaussian members in the mixed model have been two open problems in academia. There are two simple but common approaches to local optimal problems. One is to test multiple times (select different initial parameters) and take the parameters estimated by the test with the largest likelihood function as the final result. Obviously, this is easy to do operationally but the disadvantage of being time-consuming cannot be ignored. Another approach is to use other clustering methods, such as k-means, to find good initial parameters for the EM algorithm. But the problem is that k-means has its own initial sensitivities issue. Therefore, the stability and reliability of the final results are difficult to be guaranteed. However, because k-means is very simple, this method is also more common. For the problem of how to determine the number of gaussian members, there are representative deterministic methods based on Bayesian [22] Information Criterion and relevant methods based on information theory, such as Minimal Description Length, Minimal Message Length and Akaike information criterion (AIC).

Expectation-Maximization algorithm (EM), or Dempster-Laird-Rubin algorithm, is a type of optimization algorithm that iteratively performs Maximum Likelihood Estimation (MLE) [23]. This algorithm is usually used as a replacement for the Newton-Raphson method for parameter estimation of probability models that include latent variables or incomplete-data [24]. The standard calculation framework of EM algorithm alternately consists of E-step (Expectation-step) and M-step (Maximization step). The convergence of the algorithm can ensure that the iteration approaches at least the local maximum. The EM algorithm is one of the special cases of the MM algorithm (Minorize-Maximization algorithm). There are several improved versions, including the EM algorithm using Bayesian inference [22][25], the EM gradient algorithm, and the generalized EM algorithm. Because iterative rules are easy to implement and allow for flexible consideration of latent variables, the EM algorithm is widely used to handle missing measurements of data and parameter estimation for many machine learning algorithms, including Gaussian Mixture Model (GMM) and the Hidden Markov Model (HMM).

Given independent observation data $X = \{X_1, \cdots, X_N\}$, and a probability model $f(X, Z, \theta)$ containing the hidden variable $Z$ and parameter $\theta$, according to MLE theory, when the likelihood of the model is maximized the optimal single-point estimate of $\theta$ in the model is given: $\theta = arg\,\underset{\theta}{max}\,p(X|\theta)$.

$$p(X\,|\theta) = \int_a^b p(X, Z\,|\theta)dZ, \quad Z \in [a,b] \tag{5}$$

$$p(X|\theta)=\sum_{c=1}^{k}p(X,Z_c|\theta),\quad Z\in\{Z_1,\cdots,Z_k\} \tag{6}$$

Latent variables can represent missing data or any random variables that cannot be directly observed in the probability model. In the formula, the first line is the case where the latent variable is a continuous variable, and the second line is the case where the latent variable is a discrete variable. The integral or summation is also called the joint likelihood of X, Z. Without losing generality, here we use discrete variables as an example. According to the conventional method of MLE, the natural logarithm of the above formula can obtain:

$$\log p(X|\theta)=\log\prod_{i=1}^{N}p(X_i|\theta)=\sum_{i=1}^{N}\log p(X_i|\theta)=\sum_{i=1}^{N}\log[\sum_{c=1}^{k}p(X_i,Z_c|\theta)] \tag{7}$$

The above expansion considered the mutual independence of the observed data. Introduce the probability distribution q (Z) related to the latent variable, that is, the latent distribution (the latent distribution can be considered as the posterior of the latent variable to the observation data, see the E-step derivation of standard algorithms). From the Jensen inequality, the log-likelihood of the observed data has the following inequality relationship:

$$\log p(X|\theta)=\sum_{i=1}^{N}\log[\sum_{c=1}^{k}\frac{q(Z_c)}{q(Z_c)}p(X_i,Z_c|\theta)]\geq\sum_{i=1}^{N}\sum_{c=1}^{k}[q(Z_c)\log\frac{p(X_i,Z_c|\theta)}{q(Z_c)}]=L(\theta,q) \tag{8}$$

When $\theta$, $q$ makes the global maximum on the right side of the inequality, the $\theta$ obtained at least makes the left side of the inequality local maximum. Therefore, after expressing the right side of the inequality as $L(\theta,q)$, the EM algorithm has the following goal:

$$\overset{\wedge}{\theta}=\arg\max_{\theta}L(\theta,q) \tag{9}$$

$L(\theta,q)$ in the above formula is equivalent to the surrogate function in the Minorize-Maximization algorithm, which is the lower limit of the MLE optimization problem. The EM algorithm approximates the maximum of log-likelihood by maximizing the surrogate function. The EM algorithm and its improved versions are used to solve parameters of machine learning algorithms. Common examples include Gaussian Mixture Model (GMM), Probabilistic Principal Component Analysis, Hidden Markov Model (HMM) and other unsupervised learning algorithms.

## 3. Method

### 3.1. Problem Statement

A clustering problem. In the risk control case, the primary step is to do clustering for thousands of companies. It simplifies the risk control problem and increases efficiency.

We implement Vertical Federated Expectation Maximum Algorithm (Hetero-EM) to group companies into K clusters with data from different organizations.

## 3.2. The System

**Subscript**: client $m \in M$; cluster $k \in K$; sample $i \in N$; feature $j \in D$

*(1) Assumptions*

i. distribution assumption: assume each $x_i$ is sampled from one of K distribution.

ii. independent assumption: assume each $x_m$ is independent from each other.

*(2) Vertical Federated Expectation Maximum Algorithm (Hetero-EM)*

In vertical federated expectation maximum algorithm (Hetero-EM), we combine local distributions from clients to get aggerated updates. Instead of using Federated Averaging as Federated Neutral Network systems do, it integrates distributions on each round based on the independent assumption. Hence, the arbiter only knows the aggregated value of each user from each client, and it is hard to infer the exact information of a user, which inherent privacy. All in all, it is a Bayesian aggerated update that not only captures uncertainty but also make data more private from posterior sampling.

```
Algorithm Vertical Federated Expectation Maximum Algorithm for clustering
Input: M local datasets $X_m$, the number of clusters K
Output: global distribution π, local distribution θ, clusters assignment distribution φ

1) Build the training plan. The plan includes Hetero Mixture Models for the clustering problem, which uses the EM algorithm to do parameter updates among M clients.
2) Deploy the FL arbiter. The arbiter mainly distributes the FL plan to the clients, and receives local distributions, integrates the global distribution.
3) Deploy clients. Each client preprocesses their data and updates local distribution and uses encryption algorithms to encrypt sensitive data on local distributions.
4) The arbiter initiates communication. At this stage, the arbiter sends a signal to the client server, telling it the conditions required for the training plan, such as memory, capacity, size of collected data, etc.
5) Client response the arbiter. After receiving the signal from the arbiter, the client responds to the server and returns information (data size, time, etc.) to the arbiter.
6) The arbiter sends clients the training plan.
7) Start the federated training. The arbiter initializes parameters $\pi_0$ and $\theta_0$. For each round t=1, 2… T:
   Client m executes:
      $p_m\left(\mathrm{x}_i \middle| \theta_{k_{(t-1)}}\right)$
      $crypto = E\left(p_m(\mathrm{x}_i | \mu_{k_t}, {\sigma^2}_{k_t})\right)$
   Send crypto to the arbiter.
   Arbiter executes:
   For each cluster k =1, 2…K and for each sample i = 1, 2… N
   Decrypt the received crypto
      $\phi_i(\mathrm{k})_{(t)} = \frac{\pi_{k_{(t-1)}} \cdot \prod_m^M p_m\left(\mathrm{x}_i \middle| \theta_{k_{(t-1)}}\right)}{\sum_{\mathrm{k}=1}^K \pi_{k_{(t-1)}} \prod_m^M p_m\left(\mathrm{x}_i \middle| \theta_{k_{(t-1)}}\right)}$
      $\mathrm{n}_{k_{(t)}} = \sum_{i=1}^N \phi_i(\mathrm{k})_{(t)}$
      $\pi_{k_{(t)}} = \frac{\mathrm{n}_{k_{(t)}}}{n}$
   Send the aggregated distribution φ and π to M clients.
   Client executes:
   Get the aggregated distribution φ from the arbiter.
```

```
    For each cluster k =1, 2…K:
        $\theta_{k(t)} = \frac{1}{\mathrm{n}_{k(t)}} \sum_{i=1}^{N} \phi_i(\mathrm{k})_{(t)} \mathrm{x}_i$
    Use local parameters to update to the artier.
8)  The arbiter stop training under stop conditions and then send signal to clients.
```

# 4. Experiments

We conducted experiments for a clustering task on two datasets of companies from two organizations, where the data have the same sample IDs but own disjoint subsets of features.

## 4.1. Data

We train numeric Dataset A with 5000 samples and 13 features and numeric Dataset B with 5000 samples and 10 different features. These two datasets includes info about 5000 companies and open to the public @ https://gitee.com/mirrors/FATE/blob/master/examples/data/default_credit.csv. After PCA compression and scale, we have 6 features for Dataset A and 5 features for Dataset B. We have two assumptions on datasets:

i. normal distribution assumption: each $x_i$ is sampled from Normal $(\mu, \sigma^2)$.

ii. independent assumption: Normal $(\mu, \sigma^2) \propto$ Normal$_A$ $(\mu, \sigma^2) \cdot$ Normal$_B$ $(\mu, \sigma^2)$. Hence, on each round, client m executes:

$$\mu_{mk(t)} = \frac{1}{\mathrm{n}_{k(t)}} \sum_{i=1}^{N} \phi_i(\mathrm{k})_{(t)} \mathrm{x}_i \tag{10}$$

$$\sigma^2{}_{mk(t)} = \left(\frac{1}{\mathrm{n}_{k(t)}} \sum_{i=1}^{N} \phi_i(\mathrm{k})_{(t)} (\mathrm{x}_i - \mu_{i(t)}) (\mathrm{x}_i - \mu_{i(t)})^T\right) \tag{11}$$

After updating the parameters of local distribution,

$$E\left(p_m\left(\mathrm{x}_i \middle| \mu_{k_t}, \sigma^2{}_{k_t}\right)\right)$$

(12)

Where,
client $m \in M$; $M = \{A, B\}$
cluster $k \in K$; number of clusters: K = 2, 3, 5
feature $j \in D$; number of features: D = D1+D2 = 6+5 =11
sample $i \in N$; number of samples: N = 5000
homomorphic encryption algorithms E: {Paillier, RSA, BFV, CKKS}

## 4.2. Results

As shown in Fig. 4, we train for 2, 3, 5 clusters, compared with result from centered data training. The label of dataset is K=2 for good-credit companies and bad credit companies. The f1 score for hetero-EM is 0.68 while that of centered-EM is 0.35.

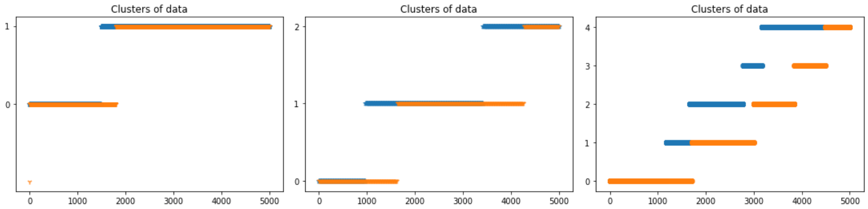


**Fig. 4.** Clusters of Data for K = 2, 3, 5

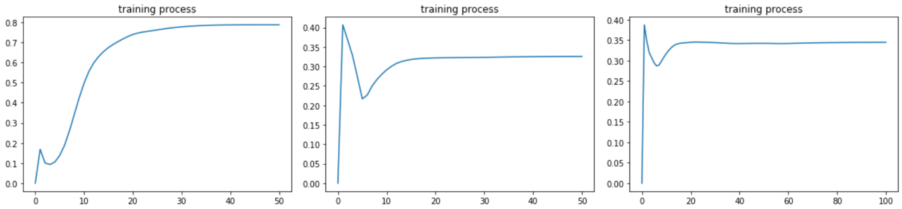


**Fig. 5.** The Probability of Cluster 0 During Training for K = 2, 3, 5

**Table 1.** Method Comparison

| | Hetero-EM(GMM) | EM(GMM) |
|---|---|---|
| Date distribution | Decentralized | Central |
| K = 2; C = 95.25% | [0.21, 0.79] | [0.23, 0.77] |
| K = 3; C = 96.35% | [0.21, 0.48, 0.31] | [0.20, 0.31, 0.49] |
| K = 5; C = 88.55% | [0.24, 0.11, 0.15, 0.30, 0.20] | [0.18, 0.18, 0.26, 0.14, 0.24] |

We evaluate the comparability $C$ by the absolute distance.

$$C = (1 - |\pi_{hetero} - \pi_{center}|) * 100\% \qquad (13)$$

**Table 2.** Homomorphic Encryption Comparison: Hetero-EM clustering for companies with K = 2; the dtype of each element is “float64”

| | PHE-A | PHE-M | FHE |
|---|---|---|---|
| Method | Paillier | RSA | CKKS |
| Keys Pair Generation | keySize: 2048bits<br>t: 1.87s | p/qSize: 1024bits<br>t: 0.338s | polyDegree: 8192bits<br>t: 0.0849s |
| Encryption | 205ms/sample | 28.9μs/sample | 10.9μs/sample |
| Decryption | 57.4ms/sample | 792μs/sample | 12.5μs/sample |
| Key Size | 2048bits | 2048bits | 4096bits |

We test different encryption algorithms on the Hetero-EM clustering for companies with K = 2. According to the table, the full homographic encryption algorithm (FHE)-CKKS is safe and the most efficient, followed by partial multiplication homographic encryption algorithm (PHE-M) and partial addition homographic encryption algorithm (PHE-M). All three encryption algorithm is safe with key pairs larger or equal to 2048 bits, but PHE-M is not efficient compared to FHE and PHE-M.

## 5. Conclusions

With improvements of artificial intelligence and arrival of the era of big data, contradictions between data sharing and data privacy are becoming a big problem. Considering of user privacy and trade secrets, it is difficult for enterprises to exchange their raw data for co-model building. In this paper, we develop a hetero-EM federated ML based on the homomorphic encryption algorithm. Each client is assumed to provide a local distribution, which is modeled through our framework. We test three encryption algorithms during communications. Besides, we design an inference strategy that allows us to integrate the global distribution in a single communication step without complex data pooling in the server. We then demonstrate the efficacy and efficiency of our Method on federated learning problems simulated from two private financial datasets.

The further works are as follows: 1) To build an improved federated learning system that is compatible to other Bayesian methods. This will make federated learning system more flexible; 2) To explore the intrinsic strategy of data spite in homomorphic encryption. This will make it more efficient and time-saving; 3) To construct a federated learning model based on Blockchain, protect user privacy and data security better.

## References

1. Li, Z.: Research on Financial Risk Control Model and Algorithm Based on Machine Learning. North China University of technology, Beijing. (2019)
2. Zhang, L.: Research on Network Loan Credit Risk Prediction Model Based on Machine Learning Algorithm. Lanzhou University, Lanzhou. (2019)
3. Gan, L.: Research on Credit Risk Prediction Model Based on Machine Learning Technology – a Case Study of an Internet Financial Company. Beijing Jiaotong University, Beijing. (2017)
4. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang, X.: Context-aware Qos Prediction with Neural Collaborative Filtering for Internet-of-Things Services. IEEE Internet of Things Journal(IoT-J), 7(5), 4532-4542. (2020)
5. Zhao, J.: Research on Credit Risk Prediction Method Based on Big Data Financial Cloud Platform. Chang'an University, Xi'an. (2017)
6. Ma, X., Gao, H., Xu, H., Bian, M.,: An IoT-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing. EURASIP Journal on Wireless Communications and Networking, 2019(1), 1-19. (2019)
7. Konečný, J., McMahan, B., Ramage, D.: Federated Optimization: Distributed Optimization Beyond the Datacenter. Mathematics, 2(1), 115. (2015)
8. Liu, M., Liao, B., Ding, L., Xiao, L.: Performance Analyses of Recurrent Neural Network Models Exploited for Online Time-Varying Nonlinear Optimization. Computer Science and Information Systems, 13(2), 691–705. (2016)
9. Lasek, P., Gryz, J.: Density-Based Clustering with Constraints. Computer Science and Information Systems, 16(2), 469–489. (2019)
10. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated Machine Learning: Concept and Applications. ACM Transactions on Intelligent Systems, 10(2), 12.1-12.19. (2019)
11. Yang, Q.: AI and Data Privacy Protection: The Way to Federated Learning. Journal of Information Security Research, 5(11), 961-965. (2019)
12. Gao, H., Kuang, L., Yin, Y., Guo, B., Dou, K.: Mining Consuming Behaviors with Temporal Evolution for Personalized Recommendation in Mobile Marketing Apps. ACM/Springer Mobile Networks and Applications (MONET), 1-16. (2020)
13. Shi, J.: Parallel Algorithm of Fully Homomorphic Encryption over Floating-Point Based on Spark. Nanjing University of Posts and Telecommunications, Nanjing. (2018)
14. He, W.: Research on Key Technologies of Privacy-Preserving Machine Learning Based on Homomorphic Encryption. University of Electronic Science and Technology of China, Chengdu. (2019)
15. Wang, J.: Homomorphic Encryption Secure Access Control Algorithm and Cloud Computing Research Based on Hyperelliptic Curve Encryption. Shaanxi University of Science and Technology, Shaanxi. (2019)
16. Xia, C.: Research of Homomorphic Encryption Technology and Application. Anhui University, Anhui. (2013)
17. Gao, H., Liu, C., Li, Y., Yang, X.: V2VR: Reliable Hybrid-Network-Oriented V2V Data Transmission and Routing Considering RSUs and Connectivity Probability. IEEE Transactions on Intelligent Transportation Systems, 1-14. (2020)
18. Meng, S.: A Survey of Machine Learning Based on Homomorphic Encryption. Computer Knowledge and Technology, 15(5), 182-183. (2019)
19. Feng, C.: Research on Related Algorithms of Fully Homomorphic Encryption. Shandong University, Shandong. (2015)
20. Qiu, T.: EM Algorithm and Its Application Research Based on Gaussian Mixture Models. University of Electronic Science and Technology of China, Chengdu. (2015)
21. Yang, X., Zhou, S., Cao, M.: An Approach to Alleviate the Sparsity Problem of Hybrid Collaborative Filtering Based Recommendations: The Product-Attribute Perspective from User Reviews. MOBILE NETWORKS & APPLICATIONS, 25(2), 376-390. (2020)

22. Zheng, Y.: Research of Large-Scale Engineering Projects' Schedule Risk Research Based on Bayesian Network. Southwest Jiaotong University, Chengdu. (2014)
23. Cao, J.: Greedy EM Algorithm Based on MapReduce Framework. Anhui University of Science and Technology, Anhui. (2018)
24. Xu, T.: The Research of EM Algorithm in Incomplete Monitoring Data Processing. Chengdu University of Technology, Chengdu. (2017)
25. A, M.: Research and Application on Naive Bayes Classification Algorithm. Dalian University of Technology, Dalian. (2014)

**Wei Ou** (corresponding author) is a lecturer at the Department of Electronic and Information Engineering, Hunan University of Science and Engineering. He received the B.S. degree from the Air Force Engineering University, in 2000, and the M.S. degree in the National University of Defense Technology in 2005, and the Ph.D. degree in the National University of Defense Technology, in 2013. His current research interests include cryptography, cyber security, AI security and Blockchain technology.

**Jianhuan Zeng** is a software engineer at the Artificial Intelligence Research Center, Qianhai Institute for Innovation Research. She received the B.S. degree from the Sun Yat-sen University, in 2017, and the M.S. degree in Columbia University in 2019. Her current research interests include cyber security and AI security.

**Zijun Guo** is currently with the Department of Electronic and Information Engineering, Hunan University of Science and Engineering. His current research interests include AI security and Blockchain technology.

**Wanqin Yan** is currently with the Department of Electronic and Information Engineering, Hunan University of Science and Engineering. Her current research interests include cyber security and AI security.

**Dingwan Liu** is currently with the Department of Electronic and Information Engineering, Hunan University of Science and Engineering. Her current research interests include cryptography and AI security.

**Dingwan Liu** is currently with the Department of Electronic and Information Engineering, Hunan University of Science and Engineering. Her current research interests include cryptography and AI security.

**Stelios Fuentes** is currently with Leicester University, UK. His research interests include distributed computing, recommendation systems, and big data.

*Received: September 23, 2019; Accepted: March 13, 2020*

Computer Science and Information Systems 17(3):835–848 https://doi.org/10.2298/CSIS190912023W

# A Novel Data-Driven Intelligent Computing Method for the Secure Control of a Benchmark Microgrid System

Shunjiang Wang[1], Yan Zhao[2], He Jiang[2], Rong Chen[1], and Lina Cao[1]

[1] State Grid Liaoning Electric Power Company Limited
110006 Shenyang, China
wangshunjiang@163.com, chenrong516@126.com, 13700007416@126.com
[2] School of Renewable Energy, Shenyang Institute of Engineering
110136 Shenyang, China
zhaoyan@sie.edu.cn, jianghescholar@163.com

**Abstract.** Microgrid is a small-scale cyber-physical system, and it generally suffers from various uncertainties. In this paper, we investigate the secure control problem of a benchmark microgrid with system uncertainties by using data-driven edge computing technology. First, the state-space function of the benchmark microgrid system is formulated, and parameter uncertainties are taken into consideration. Second, a novel data-driven intelligent computing method is derived from the model-based reinforcement learning algorithm, which only requires system data instead of system models. By utilizing this computing method, the optimal control policy can be obtained in the model-free environment. Third, the Lyapunov stability theory is employed to prove that the uncertain microgrid can be asymptotically stabilized under the optimal control policy. Finally, simulation results demonstrate the control performance can be improved by tuning the parameters in the performance index function.

**Keywords:** edge computing, microgrid system, secure control, reinforcement learning.

## 1. Introduction

Nowadays, a large-scale power system is generally composed of several distributed microgrids. With the system operating, a variety of unexpected uncertainties, which severely affect the system stability, are inevitable. Especially for microgrids, the system security issue deserves much attention. In this paper, we will study the secure control problem of a benchmark microgrid [5,13,14,23]. This benchmark microgrid consists of three main parts: power generation, loads and distributed energy storages. The power generation includes regular generation (microturbine), and supplies energy for the demands of various loads.

However, due to the intermittent power injection from photovoltaic arrays and sudden change of load demands, the unbalance between power supply and demand may occur, which will cause the frequency fluctuations and threaten the security of the entire microgrid. Thus, we incorporate distributed energy storages (electric vehicles) into this microgrid to compensate the unbalance. The system data can be measured by sensors and transmitted to the management center through the communication module. The whole microgrid is controlled by using the edge computing technology [2,3,4,24]. The schematic diagram of edge computing for the benchmark microgrid system is shown in Fig. 1.

Different from the traditional automatic control, edge computing is more like an intelligent control method which is based on computing and information, and it mainly concerns the control strategies for dispatch and optimization. In [2], by means of road networks, the problems of frequently moving vehicles and network connectivity were analyzed, and then a modified greedy algorithm for vehicle wireless communication was proposed for network optimization. In [3], a holistic framework to attack the QoS prediction was developed in the IoT environment, and authors designed a fuzzy clustering algorithm which was capable of clustering contextual information. In order to fully utilize hidden features in edge computing environment, the work [24] presented a new matrix factorization model with deep features learning via a convolutional neural network. Since the edge computing technology has such powerful merits, this paper will utilize a novel data-driven intelligent computing method for the secure control of the benchmark microgrid system.

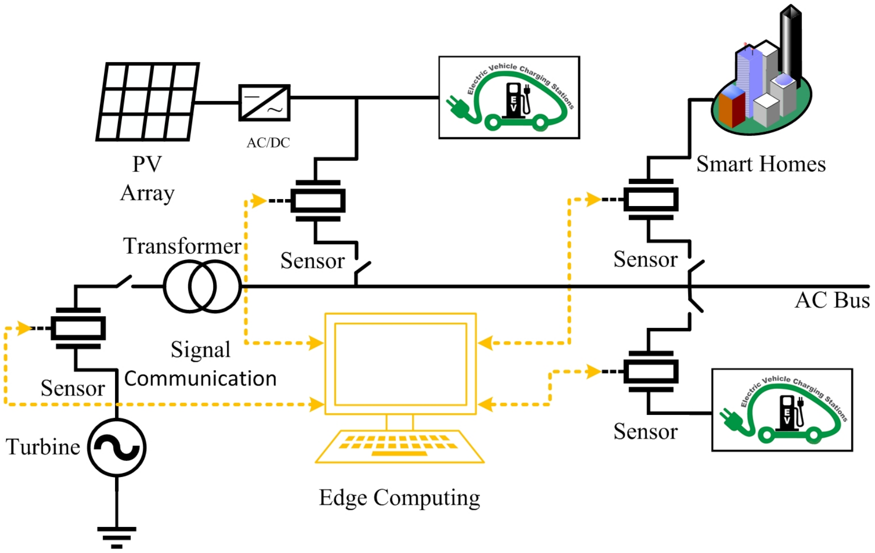


**Fig. 1.** Schematic diagram of edge computing for the benchmark microgrid system

Previous works regarding the frequency stability issues were mainly based on fuzzy control [5], sliding mode control [13,14], linear matrix inequality (LMI) approach [22] and proportion-integration-differentiation (PID) control [17]. In [12], through modeling the disturbances and parameter uncertainties, an adaptive supplementary control method was proposed for the power system frequency regulation. In [7], a novel second-order sliding mode approach for multi-area power systems was developed by means of an extended disturbance observer. In [19], a new frequency control method was designed for isolated micro-grids via double sliding mode technique. In [16], a second-order sliding mode controller was provided for the power flow control of a hybrid energy storage system. In [21], in order to eliminate the adverse effects of time delays in microgrid, a sliding mode estimation controller was developed to predict time delays and handle the disturbance of estimation errors. In [11], to deal with the uncertainties caused by renewable sources, a Takagi-Sugeno fuzzy model was constructed for the microgrid, and a corresponding sliding mode approach was designed. From aforementioned works, we can see the sliding mode control technique is a powerful tool in handling the uncertainties and disturbances.

Unfortunately, these aforementioned works were generally model-based. Due to the existence of system uncertainties, the accurate mathematical models are unavailable, and the model-based control strategies cannot be employed. Therefore, a data-driven secure control method is expected, which motivates the research of this paper.

(1) The proposed data-driven secure control method integrates reinforcement learning, optimal control theory and universal approximator.

(2) This data-driven reinforcement learning method is developed from the model-based policy iteration algorithm. Different from other reinforcement learning methods, it only requires system data instead of system models.

(3) For the secure control issues, the traditional model-based learning approaches will be invalid, because system uncertainties lead to the difficulties in obtaining the accurate mathematical models. As a result, the proposed model-free method becomes the first choice.

In this paper, we investigate the secure control problem of a benchmark microgrid with system uncertainties by using data-driven edge computing technology. The rest of this paper is arranged as follows. First, the problem formulation is given in Section 2. Second, three state-of-the-art reinforcement learning (RL) methods are introduced including policy iteration (PI), value iteration (VI) and a novel data-driven intelligent computing method in Section 3. By utilizing the data-driven computing method, the optimal control policy can be obtained in the model-free environment. Third, in Section 4, the Lyapunov stability theory is employed to prove that the uncertain microgrid can be asymptotically stabilized under the optimal control policy. In Section 5, simulation results demonstrate the control performance can be improved by tuning the parameters in the performance index function. Finally, a brief conclusion is given in Section 6.

## 2. Problem formulation

The benchmark microgrid [5,13,14,23] investigated in this paper has been introduced. Let us consider the detailed mathematical system model as below

$$\begin{aligned}
\Delta\dot{\psi}_f &= -\frac{1}{T_p}\Delta\psi_f + \frac{k_p}{T_p}\Delta\psi_t + \frac{k_p}{T_p}\Delta\psi_{v1} + \frac{k_p}{T_p}\Delta\psi_{v2} \\
\Delta\dot{\psi}_t &= -\frac{1}{T_t}\Delta\psi_t + \frac{1}{T_t}\Delta\psi_g \\
\Delta\dot{\psi}_g &= -\frac{1}{k_s T_g}\Delta\psi_f - \frac{1}{T_g}\Delta\psi_g + \frac{1}{T_g}u_1 \\
\Delta\dot{\psi}_{v1} &= -\frac{1}{T_{v1}}\Delta\psi_{v1} + \frac{1}{T_{v1}}u_2 \\
\Delta\dot{\psi}_{v2} &= -\frac{1}{T_{v2}}\Delta\psi_{v2} + \frac{1}{T_{v2}}u_3
\end{aligned} \tag{1}$$

where $\varDelta\psi_f$ denotes the frequency deviation; $\varDelta\psi_t$ is the turbine power; $\varDelta\psi_g$ is the governor position value; $\varDelta\psi_{v1}$ represents the first electric vehicle power; $\varDelta\psi_{v2}$ is the second electric vehicle power; $T_t$ denotes the time constant of turbine; $T_g$ is the time constant of governor; $T_p$ represents the time constant of power system; $T_{v1}$ is the time constant of the first electric vehicle; $T_{v2}$ denotes the time constant of the second electric vehicle; $k_p$ is the gain of power system; $k_s$ represents the speed regulation coefficient; $u_1, u_2, u_3$ are the control inputs.

Let $x = [\varDelta\psi_f, \varDelta\psi_t, \varDelta\psi_g, \varDelta\psi_{v1}, \varDelta\psi_{v2}]^T$ and $u = [u_1, u_2, u_3]^T$. The nominal system can be rewritten as

$$\dot{x} = Ax + Bu \tag{2}$$

where $A = \begin{bmatrix} -\frac{1}{T_p} & \frac{k_p}{T_p} & 0 & \frac{k_p}{T_p} & \frac{k_p}{T_p} \\ 0 & -\frac{1}{T_t} & \frac{1}{T_t} & 0 & 0 \\ -\frac{1}{k_s T_g} & 0 & -\frac{1}{T_g} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{T_{v1}} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{T_{v2}} \end{bmatrix}$ and $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{T_g} & 0 & 0 \\ 0 & \frac{1}{T_{v1}} & 0 \\ 0 & 0 & \frac{1}{T_{v2}} \end{bmatrix}$.

As is known, a microgrid system is composed of several different units and contains complex structures. With the system operating, internal faults and external disturbances may lead to the change of system structures or the deviation of system parameters. Unfortunately, system uncertainties are generally inevitable, which may affect the control performance. The system (2) with parameter uncertainties can be described by

$$\dot{x} = (A + \varDelta A)x + (B + \varDelta B)u \tag{3}$$

where $u$ is the secure control policy, which will be designed later. The parameter uncertainties $\varDelta A$ and $\varDelta B$ are bounded by $\|\varDelta A\| \leq \varDelta A_m$ and $\|\varDelta B\| \leq \varDelta B_m$, respectively.

For the microgrid system, there are two important indexes. One is for the system states, because the frequency deviation should be strictly limited. The other one is for the control inputs, because large control inputs may damage electrical elements and waste unnecessary energies. Therefore, we define the performance index function as

$$J(x(0), u) = \int_0^\infty r(x(\tau), u(\tau))d\tau \tag{4}$$

where $r(x, u) = x^T Q x + u^T R u$ with positive definite symmetric matrices $Q$ and $R$. The matrix $Q$ determines the oscillation amplitude of system states, and the matrix $R$ determines the cost of control inputs.

Given the admissible control policy $u(x)$, the value function is expressed as

$$V(x(t)) = \int_t^\infty r(x(\tau), u(x(\tau)))d\tau. \tag{5}$$

In the classical control theory, the optimal control problem is to find out a state feedback control policy which can minimize the value function. Consequently, the optimal value function can be defined as

$$V^*(x(t)) = \min_u \left( \int_t^\infty r(x(\tau), u(x(\tau)))d\tau \right). \tag{6}$$

According to the stationarity condition [18], the optimal control policy is derived by

$$u^*(x) = -\frac{1}{2}R^{-1}(B + \Delta B)^T \nabla V^*(x) \tag{7}$$

where $\nabla V^*(x) = \partial V^*(x)/\partial x$ and $V^*(x)$ should satisfy the following Hamilton-Jacobi-Bellman (HJB) equation

$$0 = r(x, u^*(x)) + \nabla V^{*T}(x)[(A + \Delta A)x + (B + \Delta B)u^*(x)]. \tag{8}$$

From the aspect of engineering, one needs to settle the HJB equation to attain the optimal control strategy. From the aspect of theory, the HJB equation is a complex partial differential equation, and it is difficult or even impossible to obtain its analytical solution. To solve the HJB equation, three algorithms will be introduced in the next section.

## 3. A brief overview of RL algorithms

In order to achieve the optimal control policy, three iterative RL algorithms including PI, VI and off-policy method will be reviewed in this section.

### 3.1. PI algorithm

Inspired by previous works [8,10,18,27], a model-based PI algorithm is given in the following Algorithm 1. By using this algorithm, one can obtain $V^*(x)$ and $u^*(x)$ as $i \to \infty$.

**Algorithm 1: PI-based RL method**
**Step 1: (Initialization)**
Let the iteration index $i = 0$.
Select a small enough computation precision $\epsilon$.
Choose an initial admissible control policy $u^{(0)}(x)$.
**Step 2: (Policy Evaluation)**
With $u^{(i)}(x)$, compute $V^{(i+1)}(x)$ by

$$0 = r(x, u^{(i)}(x)) + (\nabla V^{(i+1)}(x))^T((A + \Delta A)x + (B + \Delta B)u^{(i)}(x)). \tag{9}$$

**Step 3: (Policy Improvement)**
With $V^{(i+1)}(x)$, update the control policy $u^{(i+1)}(x)$ by

$$u^{(i+1)}(x) = -\frac{1}{2}R^{-1}(B + \Delta B)^T \nabla V^{(i+1)}(x). \tag{10}$$

**Step 4:(Termination)**
If $\left\|V^{(i+1)}(x) - V^{(i)}(x)\right\| \leq \epsilon$ on the given compact set, stop at this step;
Else, let $i = i + 1$ and go back to Step 2.

Due to the easy-to-realize structure and fast convergence, PI method is popular in the field of computer sciences. It starts from an admissible control input, and gradually approaches to the optimal solution by the steps of policy evaluation and policy improvement.

### 3.2. VI-based integral RL algorithm

The aforementioned Algorithm 1 requires initial admissible control. Although this initial condition simplifies the process of finding optimal solutions and speeds up algorithm convergence [9], it is impractical for some complex systems. Inspired by previous works [6,20], we present the following VI-based integral RL algorithm which is not limited by the initial admissible condition.

**Algorithm 2: VI-based integral RL algorithm**

**Step 1: (Initialization)**

Let the iteration index $i = 0$.

Set a computation precision $\epsilon$.

Choose an initial value function $V^{(0)}(x) \geq 0$.

**Step 2: (Policy Improvement)**

With $V^{(i)}(x)$, compute $u^{(i)}(x)$ by

$$u^{(i)}(x) = -\frac{1}{2}R^{-1}(B + \varDelta B)^T \nabla V^{(i)}(x). \tag{11}$$

**Step 3: (Policy Evaluation)**

With $u^{(i)}(x)$, update the value function $V^{(i+1)}(x)$ by

$$V^{(i+1)}(x(t)) = \int_t^{t+\varDelta t} r(x(\tau), u^{(i)}(x(\tau)))d\tau + V^{(i)}(x(t + \varDelta t)). \tag{12}$$

**Step 4:(Termination)**

If $\left\|V^{(i+1)}(x) - V^{(i)}(x)\right\| \leq \epsilon$ on the given compact set, stop at this step;

Else, let $i = i + 1$ and go back to Step 2.

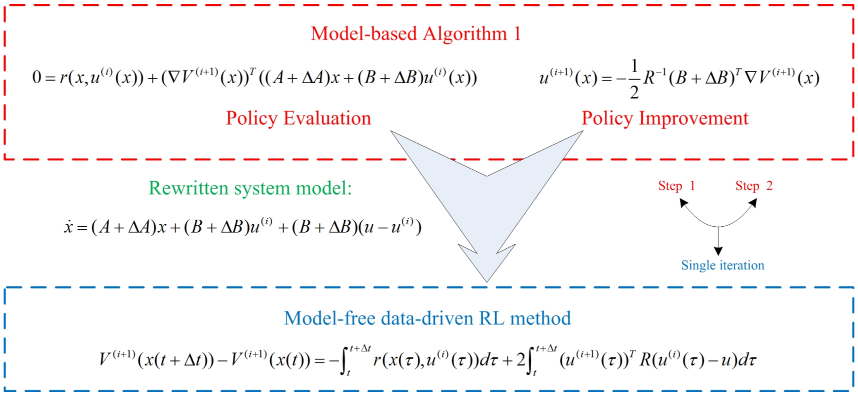

Model-based Algorithm 1
$0 = r(x, u^{(i)}(x)) + (\nabla V^{(i+1)}(x))^T ((A + \Delta A)x + (B + \Delta B)u^{(i)}(x))$
$u^{(i+1)}(x) = -\frac{1}{2}R^{-1}(B + \Delta B)^T \nabla V^{(i+1)}(x)$
Policy Evaluation
Policy Improvement
Rewritten system model:
$\dot{x} = (A + \Delta A)x + (B + \Delta B)u^{(i)} + (B + \Delta B)(u - u^{(i)})$
Step 1
Step 2
Single iteration
Model-free data-driven RL method
$V^{(i+1)}(x(t + \Delta t)) - V^{(i+1)}(x(t)) = -\int_t^{t+\Delta t} r(x(\tau), u^{(i)}(\tau))d\tau + 2\int_t^{t+\Delta t} (u^{(i+1)}(\tau))^T R(u^{(i)}(\tau) - u)d\tau$

**Fig. 2.** Derivation diagram of data-driven RL method

**Remark 1** *Algorithm 1 is completely model-based, and it requires the knowledge of both $\varDelta A$ and $\varDelta B$. Algorithm 2 is partially model-based, and it only needs the knowledge of*

*$\Delta B$. Unfortunately, the accurate models of system uncertainties $\Delta A$ and $\Delta B$ are generally unavailable. As a result, both Algorithm 1 and Algorithm 2 are not practical for the real-world engineering. A data-driven method is expected, which only needs system data instead of accurate system models. The derivation diagram of data-driven RL method is shown in Fig. 2. Two iteration steps of Algorithm 1 are combined in a single step in the data-driven RL method. Through this derivation, the system models are avoided.*

### 3.3. Data-driven RL method

In this subsection, we will introduce a model-free method, also called off-policy RL algorithm [10,15]. In order to derive this algorithm, let us rewrite the system (3) as

$$\dot{x} = (A + \Delta A)x + (B + \Delta B)u^{(i)} + (B + \Delta B)(u - u^{(i)}). \tag{13}$$

By means of (13), we have

$$\begin{aligned} \frac{dV^{(i+1)}(x)}{dt} =& (\nabla V^{(i+1)}(x))^T[(A + \Delta A)x + (B + \Delta B)u^{(i)}] \\ &+ (\nabla V^{(i+1)}(x))^T(B + \Delta B)(u - u^{(i)}). \end{aligned} \tag{14}$$

In light of (9) and (10) in Algorithm 1, Equation (14) can be rewritten as

$$\frac{dV^{(i+1)}(x)}{dt} = -r(x, u^{(i)}(x)) + 2(u^{(i+1)}(x))^T R(u^{(i)}(x) - u). \tag{15}$$

Integrating both sides of (15) on the interval $[t \;\; t + \Delta t]$ yields

$$\begin{aligned} &V^{(i+1)}(x(t + \Delta t)) - V^{(i+1)}(x(t)) \\ =& -\int_t^{t+\Delta t} r(x(\tau), u^{(i)}(\tau))d\tau + 2\int_t^{t+\Delta t} \left(u^{(i+1)}(\tau)\right)^T R(u^{(i)}(\tau) - u)d\tau. \end{aligned} \tag{16}$$

**Remark 2** *It can be seen that the policy evaluation step in Algorithm 2 is more easy-to-realize than that in both Algorithm 1 and data-driven RL method. The VI-based integral RL algorithm provides a choice to solve the optimal control issue when the admissible control is unavailable. Although Algorithm 2 contains a more relaxed initialization condition, it also has several drawbacks compared with other two algorithms. By means of initial admissible control, PI-based methods achieve faster convergence [9]. Furthermore, all the iterative control policies in the PI learning procedure are admissible, which cannot be guaranteed in the VI method [8]. The data-driven RL method is completely model-free, while Algorithm 2 is partially model-free. That is because the control input matrix $B$ is still required in Algorithm 2. Actually, when accurate system models and admissible control policies are available, the traditional PI method, i.e., Algorithm 1, is a more convenient choice. In conclusion, RL has given enough choices for different situations. The data-driven RL algorithm can cover most application demands, and the VI-based integral RL algorithm can be used without initial admissible control. The detailed performance comparisons among three iterative RL methods are shown in Fig. 3.*

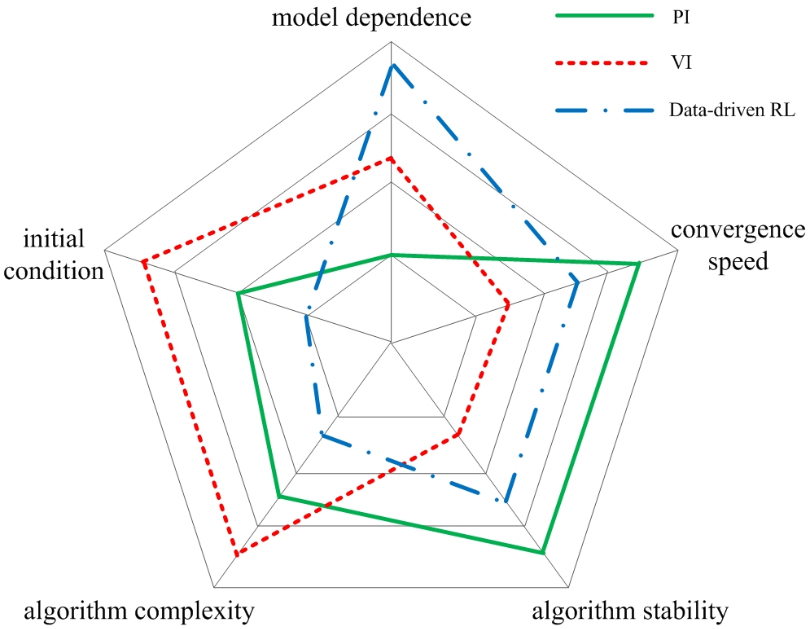


**Fig. 3.** Performance comparisons among three state-of-the-art RL methods

## 4. Algorithm implementation and stability analysis

As is known, neural network (NN) has been proved to be a powerful universal approximator. Hence, NNs are generally utilized to implement the aforementioned RL algorithms. A critic NN and an actor NN are constructed to approximate the value function and control policy, respectively. Each NN consists of two parts including tuned NN weights and NN activation functions. In this section, we will provide another implementation tool called generalized fuzzy hyperbolic model (GFHM) [1,25,26]. It also has the property of the universal approximation, and can be finally converted into the similar form as NNs.

**Definition 1** *[25,26] Let $x = [x_1, x_2, \cdots, x_n]^T$ be the model input and $y$ be the single output. $\bar{x} = [\bar{x}_1, \bar{x}_2, \cdots, \bar{x}_m]^T$ represents the generalized input, where $\bar{x}_i = x_z - d_{zj}$. $w_z$ denotes the number of transformation about $x_z$ with $z = 1, 2, \cdots, n$. $d_{zj}$ is the transformation constant for $x_z$ with $j = 1, 2, \cdots, w_z$. $m = \sum_{z=1}^{n} w_z$ denotes the total number of generalized input variables. The generalized fuzzy hyperbolic rule base should satisfy the following terms:*

1. *The pth fuzzy rule is expressed as*
   *IF $(x_1 - d_{11})$ is $F_{x_{11}}$ and $\cdots$ and $(x_1 - d_{1w_1})$ is $F_{x_{1w_1}}$ and $(x_2 - d_{21})$ is $F_{x_{21}}$ and $\cdots$ and $(x_2 - d_{2w_2})$ is $F_{x_{2w_2}}$ and $\cdots$ and $(x_n - d_{n1})$ is $F_{x_{n1}}$ and $\cdots$ and $(x_n - d_{nw_n})$ is $F_{x_{nw_n}}$,*
   *THEN $y^p = c_{F_{11}} + \cdots + c_{F_{1w_1}} + c_{F_{21}} + \cdots + c_{F_{2w_2}} + \cdots + c_{F_{n1}} + \cdots + c_{F_{nw_n}}$,*

*where $F_{x_{zj}}$ is the fuzzy set corresponding to $x_z - d_{zj}$ including $P_x$ (Positive) and $N_x$ (Negative) subsets.*

2. *$c_{F_{zj}}$ in the "THEN" part should be associated with $F_{x_{zj}}$ in the "IF" part. That is, $c_{F_{zj}}$ and $F_{x_{zj}}$ should exist or disappear at the same time.*
3. *There should be $2^m$ fuzzy rules containing all the possible combinations in both $P_x$ and $N_x$ subsets.*

The selection of the membership function is important, which should be helpful for the derivation of the following Lemma 1. Hence, the membership functions are given by

$$\begin{aligned}\mu_{P_i}(\bar{x}_i) =& e^{-\frac{1}{2}(\bar{x}_i - q_i)^2},\\ \mu_{N_i}(\bar{x}_i) =& e^{-\frac{1}{2}(\bar{x}_i + q_i)^2}\end{aligned} \tag{17}$$

where $q_i$ is a positive constant.

**Lemma 1** *[1,25] Let the membership functions be given by (17) and the generalized fuzzy hyperbolic rule base be described by Definition 1. Then, the GFHM can be derived by*

$$\begin{aligned}y(x) =& \sum_{i=1}^{m} \frac{c_{P_i} e^{q_i \bar{x}_i} + c_{N_i} e^{-q_i \bar{x}_i}}{e^{q_i \bar{x}_i} + e^{-q_i \bar{x}_i}}\\ =& \sum_{i=1}^{m} \sigma_i + \sum_{i=1}^{m} \varpi_i \frac{e^{q_i \bar{x}_i} - e^{-q_i \bar{x}_i}}{e^{q_i \bar{x}_i} + e^{-q_i \bar{x}_i}}\\ =& \theta + \rho^T \tanh(Y\bar{x})\end{aligned} \tag{18}$$

*where $\sigma_i = \frac{c_{P_i} + c_{N_i}}{2}$, $\theta = \sum\limits_{i=1}^{m} \sigma_i$, $\rho = [\varpi_1, \varpi_2, \cdots \varpi_m]^T$, $Y = diag\{q_1, q_2, \cdots, q_m\}$, $\varpi_i = \frac{c_{P_i} - c_{N_i}}{2}$ and $\tanh(Y\bar{x}) = [\tanh(q_1\bar{x}_1), \tanh(q_2\bar{x}_2), \cdots, \tanh(q_m\bar{x}_m)]^T$.*

The purpose of presenting Lemma 1 is to construct the similar expression form as NNs. According to (18), the GFHM can be further rewritten as

$$y(x) = \phi^T(x)W \tag{19}$$

where $W = [\theta, \rho^T]^T$ and $\phi(x) = [1, \tanh(q_1\bar{x}_1), \tanh(q_2\bar{x}_2), \cdots, \tanh(q_m\bar{x}_m)]^T$. Here, GFHM gets the same expression form as NNs, where $W$ can be seen as NN weights and $\phi(x)$ is similar to the NN activation function. Lemma 1 has provided the NN expression form for GFHM. Next, we will present Lemma 2 to demonstrate its property of the universal approximation.

**Lemma 2** *[1,25] For arbitrary continuous function $f(x)$ on the compact set $\Omega$ and arbitrary constant $\epsilon > 0$, there exists at least one GFHM such that $\sup\limits_{x\in\Omega} |f(x) - y(x)| < \epsilon$.*

Based on Lemma 2, the GFHM has been proved to be a universal approximator. By means of the form of (19), the GFHM provides another choice to implement RL algorithms besides NNs. Especially for dealing with nonlinear systems, the GFHM is a powerful tool in identifying nonlinear continuous functions [25].

Next, we will propose the following theorem for the stability analysis.

**Theorem 1** *If the optimal control policy $u^*(x)$ is employed, then the uncertain system (3) can be asymptotically stabilized.*

**Proof.** Choose the following Lyapunov function candidate:

$$V = V^*(x). \tag{20}$$

According to the HJB equation (8), we can obtain

$$\begin{aligned}\dot{V} =& \dot{V}^*(x) \\ =& \nabla V^{*T}(x)[(A + \varDelta A)x + (B + \varDelta B)u^*(x)] \\ =& - x^T Q x - u^{*T}(x) R u^*(x).\end{aligned} \tag{21}$$

From (21), it is obvious that $\dot{V} \leq 0$. Therefore, according to the Lyapunov stability theory, the uncertain system (3) can be asymptotically stabilized under the optimal control policy $u^*(x)$. Furthermore, by tuning the parameters $Q$ and $R$, the control performance can be improved, which will be demonstrated in the simulation part.

The proof is completed. ■

## 5. Simulation result

In order to illustrate the effectiveness of our proposed scheme, we present the following simulation result. The values of system parameters for the numerical simulation are shown in Table 1.

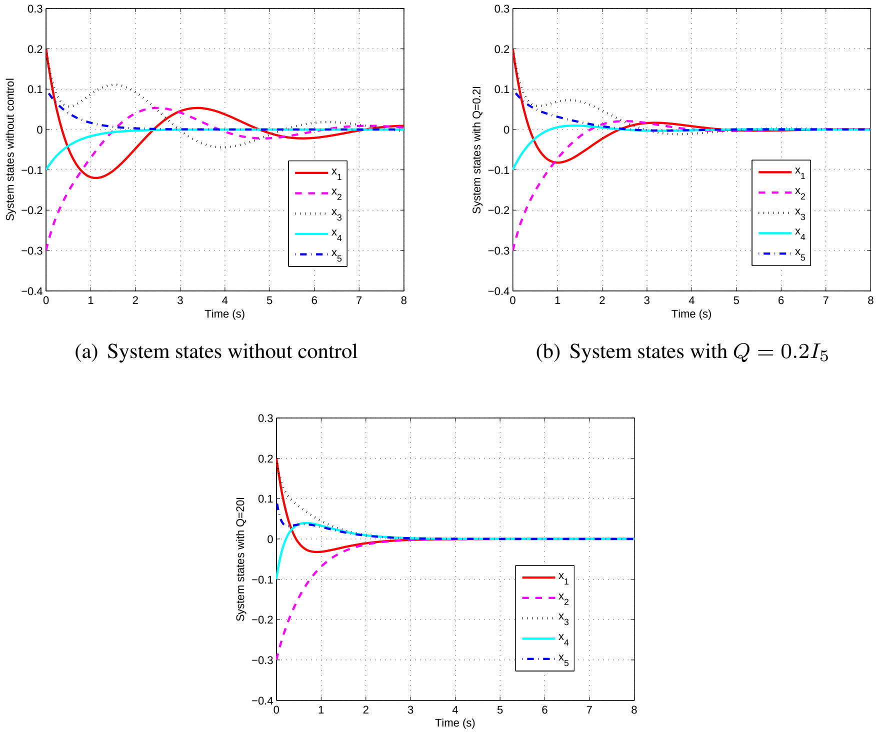
**Table 1.** Values of system parameters

| Parameters | $T_g$ | $T_t$ | $T_p$ | $T_{v1}$ | $T_{v2}$ | $k_s$ | $k_p$ |
|---|---|---|---|---|---|---|---|
| Values | 0.5 | 1 | 1 | 0.5 | 0.5 | 1 | 2 |

Through Table 1, the matrices $A$ and $B$ can be obtained. Let the system uncertainties be $\varDelta A = [0.1, 0, 0, 0, -0.2; 0, 0.1, 0, 0, 0; 0, 0, 0.2, 0, 0; 0, 0, 0, 0.2, 0; 0, 0, 0, 0, 0.2]$ and $\varDelta B = [0, 0, 0; 0, 0, 0; -0.2, 0, 0; 0, -0.2, 0; 0, 0, -0.2]$.

Set the system initial values $x(0) = [0.2; -0.3; 0.2; -0.1; 0.1]$. First, we present the simulation result without any control inputs in Fig. 4(a), where it is observed that the system without control inputs cannot be stabilized after 8 seconds.

Second, we set $Q = 0.2I_5$ and $R = I_3$, and apply the associated optimal control policy to the uncertain system. The simulation result is shown in Fig. 4(b), where we can see the system with the optimal control policy can be stabilized after 8 seconds. Third, we set $Q = 20I_5$ and $R = I_3$, and apply the associated optimal control policy to the uncertain system. The simulation result is shown in Fig. 4(c), where it can be observed that the system states under the optimal control with $Q = 20I_5$ get convergence within 4 seconds. Through the simulation results of Fig. 4, we can verify the validity of Theorem 1.

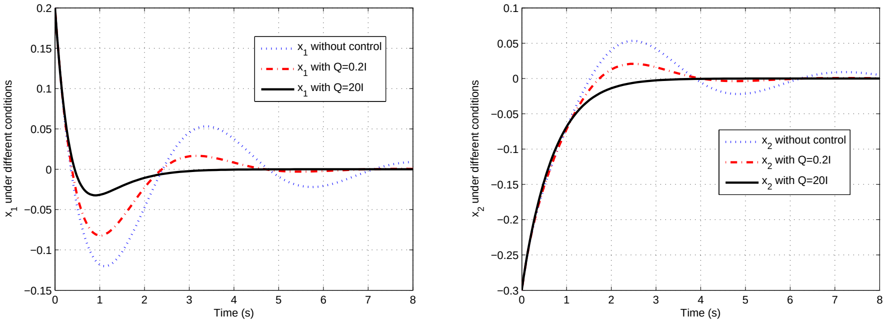


(a) System states without control

(b) System states with $Q = 0.2I_5$

(c) System states with $Q = 20I_5$

**Fig. 4.** Comparisons of control performance among different conditions



(a) System state $x_1$ under different conditions

(b) System state $x_2$ under different conditions

**Fig. 5.** System states $x_1$ and $x_2$ under different conditions

In addition, we present the system states $x_1$ and $x_2$ under different conditions in Fig. 5(a) and Fig. 5(b), respectively. From Fig. 5, we can see the system states $x_1$ and $x_2$ with $Q = 20I_5$ achieve faster convergence and better control performance than them without control or with $Q = 0.2I_5$, which implies the control performance can be improved by tuning the parameter $Q$ in the performance index function.

From the simulation results, we can see the secure control strategy relies on the optimal control theory, which implies the optimal control policy not only has the optimality but also the robustness. The matrix $Q$ in the performance index function plays an important role in handling the system uncertainties and state deviation. Once the optimal control policy is applied to the system, the security as well as the optimality will be guaranteed by tuning the matrix $Q$. Different from other secure control methods, the proposed control scheme not only concerns the robustness but also the optimality of the entire control process.

## 6. Conclusion

In this paper, the secure control problem of a benchmark microgrid with system uncertainties has been investigated by using data-driven edge computing technology. The corresponding mathematical problem formulation has been derived and established. Three state-of-the-art RL methods have been reviewed in details. The stability analysis has been presented through the Lyapunov stability theory, which indicates the uncertain microgrid can be asymptotically stabilized under the optimal control policy. Furthermore, simulation results have demonstrated the control performance can be significantly improved by tuning the parameters in the performance index function. In the future, it is expected that our proposed scheme can be applied to other information systems.

**Acknowledgments.** This work was supported by Liaoning Revitalization Talents Program (XLYC1907138), the Natural Science Foundation of Liaoning Province (2019-MS-239), the Doctoral Scientific Research Foundation of Liaoning Province (2020-BS-181), the Technology Innovation Talent Fund of Shenyang (RC190360) and Liaoning BaiQianWan Talents Program.

## References

1. Cui, Y., Zhang, H., Wang, Y., Gao, W.: Adaptive control for a class of uncertain strict-feedback nonlinear systems based on a generalized fuzzy hyperbolic model. Fuzzy Sets and Systems 302, 52–64 (2016)
2. Gao, H., Liu, C., Li, Y., Yang, X.: V2VR: Reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability. IEEE Transactions on Intelligent Transportation Systems DOI: 10.1109/TITS.2020.2983835
3. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang, X.: Context-aware QoS prediction with neural collaborative filtering for internet-of-things services. IEEE Internet of Things Journal 7(5), 4532–4542 (2020)
4. Gao, H., Duan, Y., Shao, L., Sun, X.: Transformation-based processing of typed resources for multimedia sources in the IoT environment. Wireless Networks DOI: 10.1007/s11276-019-02200-6
5. Khooban, M.H., Niknam, T., Blaabjerg, F., Dragicevic, T.: A new load frequency control strategy for micro-grids with considering electrical vehicles. Electric Power Systems Research 143, 585–598 (2017)

6. Lewis, F.L., Vrabie, D., Vamvoudakis, K.G.: Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. IEEE Control Systems 32(6), 76–105 (2012)
7. Liao, K., Xu, Y.: A robust load frequency control scheme for power systems based on second-order sliding mode and extended disturbance observer. IEEE Transactions on Industrial Informatics 14(7), 3076–3086 (2018)
8. Liu, D., Wei, Q.: Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems. IEEE Transactions on Neural Networks and Learning Systems 25(3), 621–634 (2014)
9. Luo, B., Liu, D., Huang, T., Yang, X., Ma, H.: Multi-step heuristic dynamic programming for optimal control of nonlinear discrete-time systems. Information Sciences 411, 66–83 (2017)
10. Luo, B., Wu, H.N., Huang, T.: Off-policy reinforcement learning for $H_\infty$ control design. IEEE Transactions on Cybernetics 45(1), 65–76 (2015)
11. Mi, Y., Zhang, H., Fu, Y., Wang, C., Loh, P.C., Wang, P.: Intelligent power sharing of DC isolated microgrid based on fuzzy sliding mode droop control. IEEE Transactions on Smart Grid 10(3), 2396–2406 (2019)
12. Mu, C., Tang, Y., He, H.: Improved sliding mode design for load frequency control of power system integrated an adaptive learning strategy. IEEE Transactions on Industrial Electronics 64(8), 6742–6751 (2017)
13. Mu, C., Liu, W., Xu, W., Rabiul Islam, M.: Observer-based load frequency control for island microgrid with photovoltaic power. International Journal of Photoenergy 2017, 1–11 (2017)
14. Mu, C., Tang, Y., He, H.: Observer-based sliding mode frequency control for micro-grid with photovoltaic energy integration. In: 2016 Power and Energy Society General Meeting. pp. 1–5. IEEE
15. Song, R., Lewis, F.L., Wei, Q., Zhang, H.: Off-policy actor-critic structure for optimal control of unknown systems with disturbances. IEEE Transactions on Cybernetics 46(5), 1041–1050 (2016)
16. Tabart, Q., Vechiu, I., Etxeberria, A., Bacha, S.: Hybrid energy storage system microgrids integration for power quality improvement using four-leg three-level npc inverter and second-order sliding mode control. IEEE Transactions on Industrial Electronics 65(1), 424–435 (2018)
17. Tan, W.: Unified tuning of PID load frequency controller for power systems via IMC. IEEE Transactions on Power Systems 25(1), 341–350 (2010)
18. Vamvoudakis, K.G., Lewis, F.L.: Online actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem. Automatica 46(5), 878–888 (2010)
19. Wang, C., Mi, Y., Fu, Y., Wang, P.: Frequency control of an isolated micro-grid using double sliding mode controllers and disturbance observer. IEEE Transactions on Smart Grid 9(2), 923–930 (2018)
20. Xiao, G., Zhang, H., Qu, Q., Jiang, H.: General value iteration based single network approach for constrained optimal controller design of partially-unknown continuous-time nonlinear systems. Journal of the Franklin Institute 355(5), 2610–2630 (2018)
21. Yan, H., Zhou, X., Zhang, H., Yang, F., Wu, Z.: A novel sliding mode estimation for microgrid control with communication time delays. IEEE Transactions on Smart Grid 10(2), 1509–1520 (2019)
22. Yang, F., He, J., Wang, D.: New stability criteria of delayed load frequency control systems via infinite-series-based inequality. IEEE Transactions on Industrial Informatics 14(1), 231–240 (2018)
23. Yang, J., Zeng, Z., Tang, Y., Yan, J., He, H., Wu, Y.: Load frequency control in isolated micro-grids with electrical vehicles based on multivariable generalized predictive theory. Energies 8(3), 2145–2164 (2015)
24. Yin, Y., Chen, L., Xu, Y., Wan, J., Zhang, H., Mai, Z.: QoS prediction for service recommendation with deep feature learning in edge computing environment. Mobile Networks and Applications DOI: 10.1007/s11036-019-01241-7

25. Zhang, H., Cui, Y., Wang, Y.: Hybrid fuzzy adaptive fault-tolerant control for a class of uncertain nonlinear systems with unmeasured states. IEEE Transactions on Fuzzy Systems 25(5), 1041–1050 (2017)
26. Zhang, H., Zhang, J., Yang, G.H., Luo, Y.: Leader-based optimal coordination control for the consensus problem of multiagent differential games via fuzzy adaptive dynamic programming. IEEE Transactions on Fuzzy Systems 23(1), 152–163 (2015)
27. Zhao, D., Zhang, Q., Wang, D., Zhu, Y.: Experience replay for optimal control of nonzero-sum game systems with unknown dynamics. IEEE Transactions on Cybernetics 46(3), 854–865 (2016)

**Shunjiang Wang** received the Ph.D. degree in computer application technology in 2019 from University of Chinese Academy of Sciences, Shenyang, China. He is now working at Power Dispatching and Control Center, State Grid Liaoning Electric Power Company Limited. His current research interests include power system automation and clean energy accommodation.

**Yan Zhao** received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2008. He is currently a professor with the School of Renewable Energy, Shenyang Institute of Engineering, Shenyang. His current research interests include sliding mode control and integrated energy system.

**He Jiang** received the Ph.D. degree in control theory and control engineering in 2019 from Northeastern University, Shenyang, China. He is now working at School of Renewable Energy, Shenyang Institute of Engineering. His current research interests include adaptive dynamic programming, sliding mode control, multi-agent system control and their industrial applications.

**Rong Chen** is now working at State Grid Liaoning Electric Power Company Limited. Her current research interests include power system automation and clean energy accommodation.

**Lina Cao** is now working at State Grid Liaoning Electric Power Company Limited. Her current research interests include economic dispatch and automatic control of power systems.

*Received: September 12, 2019; Accepted: June 8, 2020.*

Computer Science and Information Systems 17(3):849–865 https://doi.org/10.2298/CSIS190925024B

# A Recommendations Model with Multiaspect Awareness and Hierarchical User-Product Attention Mechanisms

Zhongqin Bi[1], Shuming Dou[2], Zhe Liu[3], and Yongbin Li[4]

[1] College of Computer Science and Technology, Shanghai University of Electric Power,
Shanghai, China
zqbi@shiep.edu.cn
[2] College of Computer Science and Technology, Shanghai University of Electric Power,
Shanghai, China
d2313776747@163.com
[3] State Grid Shanghai Electric Power Research Institute, Shanghai, China
liuzheacyy@163.com
[4] College of Computer Science and Technology, Shanghai University of Electric Power,
Shanghai, China
lybin40000@163.com

**Abstract.** Neural network methods have been trained to satisfactorily learn user/product representations from textual reviews. A representation can be considered as a multiaspect attention weight vector. However, in several existing methods, it is assumed that the user representation remains unchanged even when the user interacts with products having diverse characteristics, which leads to inaccurate recommendations. To overcome this limitation, this paper proposes a novel model to capture the varying attention of a user for different products by using a multilayer attention framework. First, two individual hierarchical attention networks are used to encode the users and products to learn the user preferences and product characteristics from review texts. Then, we design an attention network to reflect the adaptive change in the user preferences for each aspect of the targeted product in terms of the rating and review. The results of experiments performed on three public datasets demonstrate that the proposed model notably outperforms the other state-of-the-art baselines, thereby validating the effectiveness of the proposed approach.

**Keywords:** recommendation, long short-term memory, attention mechanism, deep learning, rating prediction.

## 1. Introduction

The explosive growth in the amount of available digital information and the increasing number of Internet users has created a potential challenge pertaining to information overload, which hinders users from gaining timely access to their products of interest on the Internet. To alleviate the inconvenience caused by such information overload, recommendation systems are often employed. Recommendation systems do not require users to provide explicit information; instead, the users' interests are modeled by analyzing the users' historical behavior. Thus, recommendation systems can proactively recommend information that meets the users' interests and needs. In this manner, recommendation systems not only relieve the pressure of customer information overload but also help the platform optimize services and generate greater commercial benefits.

Recommendation algorithms have been widely integrated into many business application systems, such as the Netflix online video recommendation system or the Amazon online shopping mall. In the early stages of development, recommendation systems were based on demography [26], which was effective at providing real-time responses, although its reliability and interpretability were low. As data regarding users accumulates, it is more desirable to provide recommendations based on the users' interaction records. Recommender systems are generally based on two mechanisms: collaborative filtering (CF) [28] and content-based recommendation [4]. These methods can be used to realize enhanced recommendation performance and interpretability; however, both approaches have limitations due to data sparsity and cold start. To overcome the limitations associated with using any one specific recommendation method, hybrid recommendation systems have been developed [24, 32, 36] that combine several recommendation technologies to obtain a better recommendation effect.

In this paper, we propose a multiaspect awareness model with a hierarchical user-product attention (AHUPA) framework to provide recommendations. The proposed approach can adaptively learn both user preferences and product characteristics and can accurately capture the varying attention that a user gives to each aspect of different products. Compared with traditional collaborative filtering recommendation algorithms, this model possesses several unique advantages. First, a review encoder and a user/product encoder are constructed to learn better review representations and user/product representations from words and reviews, respectively. The attention network can capture the key words and reviews, making the extracted representations more reliable. Second, considering the significance of the overall ratings, we combine the user-product attention vectors learned from the encoders by using the user and product embeddings based on the IDs as the final representations for deep learning. Finally, to determine the attentive interaction of the users' and products' final latent factors corresponding to various aspects, we design a rating encoder with adaptive multiaspect awareness for each user-product pair. To sum up, the main contributions of this work are as follows.

- We propose a multiaspect awareness model for recommendations. With hierarchical user-product attention mechanisms, respectively, two representations are generated, which are concatenated for further representation learning.
- We introduce an attention network to capture the varying attention vectors of each specific user-product pair.
- We conduct comprehensive experiments on three public datasets to comparatively evaluate and demonstrate the effectiveness of the proposed approach.

The remainder of this paper is organized as follows. Section 2 describes the related works, Section 3 presents the preliminary definitions, and Section 4 describes the AHUPA model in detail. Experiments involving the training of the AHUPA model are presented and the advantages of the AHUPA model compared to other state-of-the-art methods are discussed in Section 5. Section 6 presents conclusions and describes future research directions.

## 2. Related Work

Learning accurate user and product representations is highly important for recommender systems. Many existing recommendation methods learn user and product representations based on the ratings that users give to products. Matrix factorization (MF) [11, 19, 27, 29] has become the most popular collaborative filtering approach[28]. Depending on the user-product rating matrix, the user interest and product characteristics are expressed as potential factor vectors in the public potential space, while the interaction rating between users and products is expressed in terms of the inner product. However, a rating only indicates a user's overall preference for a product, and the interpretability of the indication is low. For example, a user might assign a high score to a slim computer, but this information may not be supported by the overall rating. Consequently, MF methods cannot achieve fine-grained modeling of user preferences on the various product aspects, leading to unexplainable recommendations. Moreover, because most of the entries are missing, a rating matrix is usually extremely sparse, which leads to cold start problems and low recommendation accuracy [9, 10, 34].

To address these issues, several researchers [5, 12, 23] have combined auxiliary information with ratings to improve the recommendation performance. As supplementary information for ratings, reviews contain rich semantics that can be mined to identify user preferences and product characteristics [39]. Different approaches have been developed to utilize textual reviews for recommendations. such as HFT[23], TriRank [21], EFM [40], and LDA [3]. These topic-based methods integrate topic models in their frameworks to automatically identify aspect information in reviews for better user and product modeling. Specifically, EFM, TriRank and LDA have explicitly claimed that they can provide explanations for recommendations. Although those methods have achieved better performances than have MF methods using only ratings, the features are simple extractions of words or phrases from the texts, which changes the integrity of the reviews and may distort their original meanings. Therefore, these shallow feature extraction methods cannot accurately model user and product representations.

In recent years, with the help of the powerful feature extraction ability of deep learning models, some studies have also tried to combine different neural network structures with collaborative filtering to learn feature interactions. In [15], He et al. presented a neural collaborative filtering (NCF) framework to learn the nonlinear interactions between users and items. Later, researchers attempted to incorporate critical information into the NCF model to improve the recommendation performance. Zheng et al. [41] proposed a deep cooperative neural network (DeepCoNN) that uses convolutional neural networks to learn the user and product representations from reviews; this model led to significantly improved recommendations. Chen et al. [6] proposed a neural attentional regression model with review-level explanations (NARRE), which took into account the effectiveness of the comments while predicting the score and explains the recommendation results at the level of comment information. Zhang et al. [38] developed a neural user-item coupling model (CoupledCF) to realize joint learning in terms of explicit and implicit coupling and used the review information to enhance the user-product coupling relationships. These methods first train unsupervised neural networks to obtain word embedded representations and then integrate deep neural networks with rating information to achieve good recommendation performance. However, they overlook the fact that the user's attention preferences regarding product aspects vary for different products, which may result in weak recommendations. User preferences can be understood as multiple aspects of the attention vector, which changes according to different products. Consequently, users have different preferences for the functions of different products of the same category; for example, in the computer category, in addition to focusing on basic features, a user will expect an expensive computer to have higher resolution and rendering capabilities than a cheap computer.

Attention mechanisms have achieved immense success in speech recognition, computer vision, natural language processing and exploration of the hierarchical attention mechanism [1, 2, 22, 31, 35, 37]. The key idea of soft attention is to learn to assign attentive weights (normalized by summing to 1) for a set of features: higher (lower) weights indicate that the corresponding features are informative (less informative) for the end task, which learns and determines the focus areas and enables the model to focus on the most effective information with limited resources. In the field of recommendation systems, researchers proposed the deep knowledge aware network (DKN) [33] and introduced an attention network to learn user representations based on the users' news-click records. Chen et al. [6] modeled the usefulness of reviews by using review-level attention to enhance the learning of both the user and product representations. In this paper, we propose a multiaspect awareness attention model that learns the user preference weights over various product aspects and then guides the interaction between user preferences and product characteristics for different aspects.

## 3. Preliminaries

### 3.1. Long Short-term Memory

The long short-term memory (LSTM) [16] model is a sequential convolutional network derived from a recurrent neural network (RNN) that solves the gradient disappearance and gradient explosion problems endemic to RNNs by utilizing a cell state and a gate mechanism. LSTM is widely used in long-sequence text modeling tasks due to its excellent performance when learning and memorizing long sequential data.

Three gate structures are used to control the state flow in the LSTM cell. In each time step $t$, given an input sequence vector $\boldsymbol{x}_t$, the current cell state $\boldsymbol{c}_t$ and hidden state $\boldsymbol{h}_t$ can be updated with the previous cell state $\boldsymbol{c}_{t-1}$ and hidden state $\boldsymbol{h}_{t-1}$ as follows:

$$\begin{bmatrix} \boldsymbol{i}_t \\ \boldsymbol{f}_t \\ \boldsymbol{o}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \end{bmatrix} \left( \boldsymbol{W} \left[ \boldsymbol{h}_{t-1}; \boldsymbol{x}_t \right] + \boldsymbol{b} \right), \tag{1}$$

$$\hat{\boldsymbol{c}}_t = \tanh \left( \boldsymbol{W}_c \left[ \boldsymbol{h}_{t-1}; \boldsymbol{x}_t \right] + \boldsymbol{b}_c \right), \tag{2}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \hat{\boldsymbol{c}}_t, \tag{3}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t), \tag{4}$$

where $\boldsymbol{i}_t$, $\boldsymbol{f}_t$ and $\boldsymbol{o}_t$ denote the input gate, forget gate and output gate in the gate mechanism, respectively; $\sigma$ denotes the logistic sigmoid function; $\tanh$ is the hyperbolic tangent activation function; and $\odot$ represents elementwise multiplication. Intuitively, the input gate $\boldsymbol{i}_t$ determines the amount of information the current $\boldsymbol{x}_t$ reserves for $\boldsymbol{c}_t$; the forget gate $\boldsymbol{f}_t$ determines the amount of the previous cell state $\boldsymbol{c}_{t-1}$ saved to the current $\boldsymbol{c}_t$; and the output gate $\boldsymbol{o}_t$ determines the amount of $\boldsymbol{c}_t$ passed to the output $\boldsymbol{h}_t$ of the current state.

In practice, because the output at the current moment is related not only to the previous state but also to the future state, the prediction must be performed while considering the combination of the forward and backward inputs. A more common approach is to apply a bidirectional LSTM to simulate the text semantics from the forward and backward inputs. For the sequence vectors $[\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_T]$, the forward and backward LSTMs read the sequences from $\boldsymbol{x}_1$ to $\boldsymbol{x}_T$ and from $\boldsymbol{x}_T$ to $\boldsymbol{x}_1$, respectively. Subsequently, the forward hidden state $\overrightarrow{\boldsymbol{h}_t}$ and backward hidden state $\overleftarrow{\boldsymbol{h}_t}$ are connected to represent the current hidden layer state $\mathbf{h}_t$, i.e., $\boldsymbol{h}_t = \left[\overrightarrow{\boldsymbol{h}_t}\,;\,\overleftarrow{\boldsymbol{h}_t}\right]$, which learns the whole sequence information corresponding to the target word $w_{it}$.

### 3.2. Attention Mechanism

The attention mechanism refers to human attention and modes of thought. Humans can rapidly screen valuable information from a large amount of content. The attention mechanism uses an attention calculation method to extract the weight of the key information, and it is widely used in various deep learning tasks, such as natural language processing, image classification and speech recognition. In particular, in the field of text processing, the attention mechanism can model the hierarchical semantics of words, sentences, and documents.

For the input information vectors $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]$, the additive model is used to score the attention to select the important information in $\boldsymbol{X}$. The soft selection mechanism is adopted to capture the weight distribution of the input information as follows:

$$e_i = \boldsymbol{v}^\top \tanh\left(\boldsymbol{W}\boldsymbol{x}_i + \boldsymbol{b}_i\right), \tag{5}$$

$$\alpha_i = \frac{\exp\left(e_i\right)}{\sum_{j=1}^{N} \exp\left(e_j\right)}, \tag{6}$$

where the probability vector of $\alpha_i$ is termed the distribution of attention and $\boldsymbol{W}$ and $\boldsymbol{v}$ are learnable network parameters. The input information is aggregated by weighted averaging as follows:

$$\boldsymbol{s} = \sum_{i=1}^{N} \alpha_i \boldsymbol{x}_i. \tag{7}$$

## 4. Methods

In this section, we describe the proposed AHUPA model in detail. The framework of AHUPA is illustrated in Figure 1. The proposed model involves three major modules: a review encoder, which learns the representations of the reviews from words; the user/product encoder, which learns the representations of the users and products from reviews; and the rating encoder, which captures the user's attention weight regarding various product aspects. Next, we introduce each module in detail.

### 4.1. Review Encoder

Given a review with the words $[w_{i1,}w_{i2,}..., w_{it}]$, $t \in [1, T]$, we first embed the reviews to the vector sequence $[\boldsymbol{e}_{i1,}\boldsymbol{e}_{i2,}..., \boldsymbol{e}_{it}]$ through an embedding matrix $\boldsymbol{E} \in R^{\mathrm{V}\times\mathrm{D}}$, where

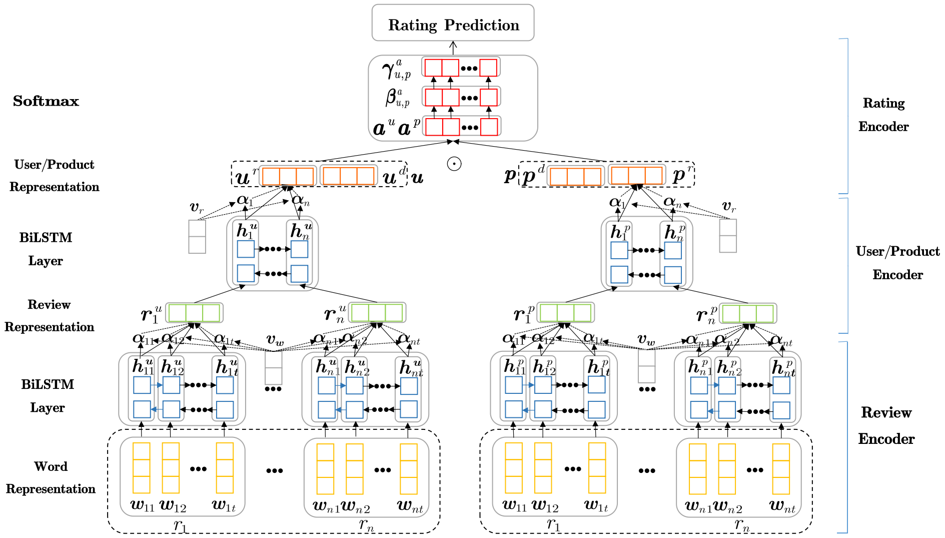


**Fig. 1.** Architecture of the proposed AHUPA model

V and D represent the vocabulary size and word embedding dimension, respectively. We obtain the word embedding $\boldsymbol{E}$ by training an unsupervised word2vec [27] model fine-tuned during training. We use a bidirectional LSTM to extract the contextual information in both directions for the words; specifically, this LSTM contains both a forward and backward LSTM that read a review in forward and reverse order, respectively:

$$\overrightarrow{\boldsymbol{h}_{it}} = \overrightarrow{\text{LSTM}}\left(\boldsymbol{e}_{it}\right), t \in [1, T], \tag{8}$$

$$\overleftarrow{\boldsymbol{h}_{it}} = \overleftarrow{\text{LSTM}}\left(\boldsymbol{e}_{it}\right), t \in [T, 1]. \tag{9}$$

We obtain the word output representation $\boldsymbol{h}_{it}$ by concatenating the forward hidden state $\overrightarrow{\boldsymbol{h}_{it}}$ and the backward hidden state $\overleftarrow{\boldsymbol{h}_{it}}$, i.e., $\boldsymbol{h}_{it} = \left[\overrightarrow{\boldsymbol{h}_{it}}; \overleftarrow{\boldsymbol{h}_{it}}\right]$, which learns the review information around the target word $w_{it}$.

From the product perspective, not all words equally reflect the product's characteristics. Hence, we introduce a word-level attention mechanism to help the models select the important words based on the contextual information and build more informative word representations. The attention weight of the $t$-th word in the $i$-th review is computed as follows:

$$u_{it} = \boldsymbol{v}_w^T \tanh\left(\boldsymbol{W}_w \boldsymbol{h}_{it} + \boldsymbol{b}_w\right), \tag{10}$$

$$\alpha_{it} = \frac{\exp\left(u_{it}\right)}{\sum_{t=1}^{T} \exp\left(u_{it}\right)}. \tag{11}$$

where $\boldsymbol{v}_w \in R^{N_w}$ and $\boldsymbol{b}_w \in R$ denote the attention network parameters, and $\alpha_{it}$ represents the relative importance of the $t$-th word in the attention network evaluation. The final representation of the $i$-th review is a weighted sum of the word level representations in a view as follows:

$$\boldsymbol{r}_i = \sum_{t=1}^{T} \alpha_{it} \boldsymbol{h}_{it}. \tag{12}$$

## 4.2. User/Product Encoder

We build the representations of the users or products based on the representations of their reviews in a similar manner. We first use a bidirectional LSTM to encode the reviews:

$$\overrightarrow{\boldsymbol{h}_i} = \overrightarrow{\text{LSTM}}\left(\boldsymbol{r}_i\right), i \in [1, L], \tag{13}$$

$$\overleftarrow{\boldsymbol{h}_i} = \overleftarrow{\text{LSTM}}\left(\boldsymbol{r}_i\right), i \in [L, 1], \tag{14}$$

where $L$ is the number of reviews of a user or product. We concatenate $\overrightarrow{\boldsymbol{h}_i}$ and $\overleftarrow{\boldsymbol{h}_i}$ to obtain an output representation of the $i$-th review, i.e., $\boldsymbol{h}_i = \left[\overrightarrow{\boldsymbol{h}_i}\,;\,\overleftarrow{\boldsymbol{h}_i}\right]$. $\boldsymbol{h}_i$ summarizes the neighbor reviews around the $i$-th review.

Different reviews contribute differently to the modeling of a user or product. Therefore, at the review level, we use an attention framework to help the proposed model distinguish informative reviews from less informative reviews, thereby generating representations of users or products that are better at obtaining user preferences and product characteristics. The attention weight of the $i$-th review is computed as follows:

$$u_i = \boldsymbol{v}_r^T \tanh\left(\boldsymbol{W}_r \boldsymbol{h}_i + \boldsymbol{b}_r\right), \tag{15}$$

$$\alpha_i = \frac{\exp\left(u_i\right)}{\sum_{i=1}^{L} \exp\left(u_i\right)}, \tag{16}$$

where $\boldsymbol{v}_r \in R^{N_s}$ and $\boldsymbol{b}_r \in R$ are the attention network parameters, and $\boldsymbol{W}_r$ represents the weight matrices. The final representation of the user or product is the summation of the contextual representations of the reviews weighted by their attention weights:

$$\boldsymbol{u} = \sum_{i=1}^{L} \alpha_i \boldsymbol{h}_i. \tag{17}$$

## 4.3. Rating Encoder

In this section, we describe the attention network used to determine the user's attention weight for different aspects of the product.

Although a large amount of valuable information can be learned from reviews, several of the underlying characteristics of the users and products can be inferred from the ratings instead of from the reviews. Therefore, the module input is divided into two parts. In one part, the review encoder and the user/product encoder obtain the user preferences or product characteristics from the reviews. The other part converts the user and product IDs into a dense vector through the embedding layer. Next, the ID-based embedded features and review-based features of the users and products are used as the input to the next layer.

Feature fusion can be conducted to improve model learning. In previous works, different fusion methods have had different effects on model performances. In this work, we select the connection method to perform feature fusion. The final user representation is a combination of the user preference $\boldsymbol{u}^r$ and user density vector $\boldsymbol{u}^d$, as follows:

$$\boldsymbol{u} = \left[\boldsymbol{u}^r; \boldsymbol{u}^d\right]. \tag{18}$$

Similarly, the final product representation is a combination of the product preference and product density vector:

$$\boldsymbol{p} = \left[\boldsymbol{p}^r; \boldsymbol{p}^d\right]. \tag{19}$$

Moreover, to further improve the model's learning ability, we add a fully connected layer after the fusion step and adopt the nonlinear ReLU activation function [13].

The core of the rating encoder is to determine the specific attention weight of user $u$ regarding one aspect $k$ of product $p$, $k \in [0, K]$, where K is the dimension of the latent vectors. Although the user preferences and product characteristics can be clearly reflected by the reviews, certain latent characteristics can be obtained by observing the rating patterns. Thus, we rely on reviews and ratings to capture users' preferences for various aspects of a product. Subsequently, the attention weight based on reviews is computed as follows:

$$\boldsymbol{r} = \left[\boldsymbol{u}^r; \boldsymbol{p}^r\right], \tag{20}$$

$$\boldsymbol{\gamma}_{u,p} = \boldsymbol{v}^T \mathrm{Re}LU\left(\boldsymbol{W}_a \boldsymbol{r} + \boldsymbol{b}_a\right), \tag{21}$$

$$\gamma_{u,p,k}^a = \frac{\exp\left(\gamma_{u,p,k}\right)}{\sum_{j=1}^{K} \exp\left(\gamma_{u,p,k}\right)}. \tag{22}$$

where $\boldsymbol{W}_a$ and $\boldsymbol{b}_a$ denote the weight matrix and bias vector, respectively. The attention weight $\beta_{u,p,k}^a$ based on ratings is computed in a similar manner.

We set $\boldsymbol{a}^u$ and $\boldsymbol{a}^p$ as the feature vectors of the user and product through the feature fusion, respectively. In addition, two attention factors are introduced to the interaction between the user and product that allow the user to adjust the degree of attention to different products according to the attention weight distributions $\boldsymbol{\gamma}_{u,p}^a$ and $\boldsymbol{\beta}_{u,p}^a$. The output of the attention interaction fusion can then be used as a rating feature as follows:

$$\boldsymbol{z}_{u,p}^a = \boldsymbol{\gamma}_{u,p}^a \odot \boldsymbol{\beta}_{u,p}^a, \tag{23}$$

$$\boldsymbol{F} = \boldsymbol{z}_{u,p}^a \odot \left(\boldsymbol{a}^u \odot \boldsymbol{a}^p\right), \tag{24}$$

where $\boldsymbol{z}_{u,p}^a \in R^K$ denotes a user's attention vector to a product, and $\odot$ is the elementwise product. We assume that the user's attention is distributed in k dimensions. According to the relation $f_k = z_{u,p,k}^a \cdot a_k^u \cdot a_k^p$, an attentional weight $z_{u,p,k}^a$ adjusts the interaction depth of the user-product pair under a factor k, which indicates that $z_{u,p,k}^a$ reflects the importance of factor k in the relationship between the user and the product.

In recommendation systems, the system recommends products based on the user's predicted rating of a product. In the proposed model, a linear layer and a softmax layer are used to project the score representation $\boldsymbol{F}$ into a score distribution of N classes:

$$q = \mathrm{softmax}\left(\boldsymbol{W}\boldsymbol{F} + \boldsymbol{b}\right). \tag{25}$$

Finally, we use cross entropy as the loss function to optimize the model parameters and minimize the difference between the actual and predicted rating:

$$L = -\sum_{d \in T} \sum_{n=1}^{N} q_n^g(\boldsymbol{F}) \cdot \log\left(q_n(\boldsymbol{F})\right). \tag{26}$$

where $q_n^g$ is the probability of the rating score $n$, with the ground truth being 1 and the other values being 0. $T$ represents the training set.

## 5. Experiments and Evaluation

### 5.1. Experimental Settings

We evaluated the effectiveness of the proposed model on three publicly accessible subsets from the Amazon collection[5] [14], i.e., **Patio, Lawn and Garden**, **Video Games**, and **CDs and Vinyl**. The dataset statistics are listed in Table 1. Similar to the approach reported in [14], we preprocessed all the datasets by retaining only users and products with at least 5 reviews. Moreover, we removed the punctuation, stopwords, and infrequent symbols (those appearing less than 10 times) from each user/product review. The ratings range in these datasets is [1, 5].

**Table 1.** Statistics of the evaluation datasets

| Datasets | #users | #products | #reviews |
|---|---|---|---|
| Patio, Lawn and Garden | 1,686 | 962 | 13,272 |
| Video Games | 24,303 | 10,672 | 231,780 |
| CDs and Vinyl | 75,258 | 64,443 | 1,097,592 |

We pretrained a word2vec [25] model to initialize the word embedding matrix $\boldsymbol{E}$. We randomly initialized all the weight parameters to conform to the uniform distribution $U(-0.01{,}0.01)$. To speed up training, we limited each user and product to at most 10 reviews, where every review has no more than 60 words. We used the Adam optimizer [17] as the optimization algorithm and set the initial learning rate to 0.005; the dropout technique [30] was used to prevent overfitting, and the dropout ratio was 0.5. We randomly selected $80\%$ of the user-product pairs in each dataset for training, $10\%$ for validation, and $10\%$ for testing. The optimal parameters were selected based on the model's performance on the validation set. Root mean square error (RMSE) was used to evaluate model performances. The RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y^{\hat{(i)}} - y^{(i)}\right)^2}, \tag{27}$$

where $\boldsymbol{N}$ is the number of samples, $y^{\hat{(i)}}$ and $y^{(i)}$ are the predicted rating score and the gold rating score respectively of the $i$-th user-product pair.

[5] http://jmcauley.ucsd.edu/data/amazon

### 5.2. Baselines

We compared the proposed AHUPA model with several baseline methods in terms of rating predictions. The methods used for comparison included the following:

- **PMF** [27]: Probabilistic matrix factorization that provide recommendations based on ratings via matrix factorization.
- **SVD++** [18]: A recommendation method based on a rating matrix via an SVD and similarities between items.
- **HFT** [23]: HFT models ratings and review texts using MF and LDA, respectively. Then, an exponential transform function is used to associate the latent topics and latent factors for rating prediction.
- **A$^3$NCF** [8]: A benchmark deep neural network that combines ratings with reviews to predict ratings. This model introduces a topic model to extract the user preferences and product characteristics from the review text.
- **CoupledCF** [38]: A coupled learning model that combines ratings and reviews to provide recommendations based on non-IID values. A doc2vec [20] framework is trained to obtain the user or product characteristics based on the reviews.
- **NARRE** [6]: A neural attention regression model with review level interpretation that uses attention mechanisms to simulate the informality of the recommended comments.

### 5.3. Model Comparisons

**Table 2.** Results obtained using different methods on different datasets. RMSE (lower values are better) is the evaluation metric

| Models | Patio, Lawn and Garden | Video Games | CDs and Vinyl |
|---|---|---|---|
| PMF | 1.1329 | 1.2052 | 1.1040 |
| SVD++ | 0.9185 | 0.9543 | 0.8949 |
| HFT | 0.9073 | 0.9475 | 0.8920 |
| A$^3$NCF | 0.8855 | 0.9402 | 0.8871 |
| CoupledCF | 0.8864 | 0.9345 | 0.8827 |
| NARRE | 0.8725 | 0.9249 | 0.8792 |
| AHUPA | 0.8676 | 0.9181 | 0.8699 |

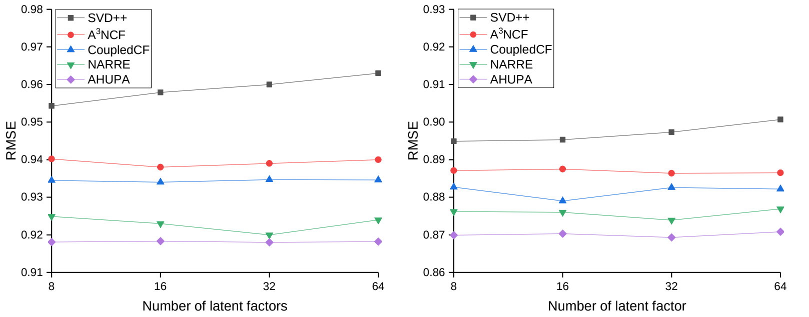
The rating prediction results of the different methods are presented in table 2. In addition, Figure 2 shows the performances of the partially adopted methods under different numbers of predictive factors. Considering space limitations, we show only the results on the two relatively larger datasets. For the MF methods (PMF and SVD++), the number of predictive factors is equal to the number of latent factors. Due to the weak performance of PMF, it is omitted in Figure 2 to better highlight the performance differences among the other methods. Based on these findings, the following observations can be made.

First, the methods that incorporate reviews (i.e., HFT, CoupledCF, NARRE, A$^3$NCF, AHUPA) exhibit a better performance than do the collaborative filtering models (PMF, SVD++) that consider only the rating matrix as the input. This is not surprising, because review information is complementary to ratings and contains rich information about user preferences and product characteristics. Therefore, better-quality modeling increases the accuracy when learning user preferences and product features and leads to a better rating prediction result.

Second, compared to the HFT model, which utilizes traditional matrix factorization techniques, the methods that utilize deep learning technology (i.e., CoupledCF, NARRE, AHUPA) demonstrate superior performances among the methods that exploit reviews. This phenomenon likely occurs because deep learning can model users and products in a nonlinear [15] way. Previous works [37] have also demonstrated that methods based on neural networks capture semantic meaning better than topic models such as LDA [3] when analyzing textual information.

Third, as shown in Table 2, our method AHUPA achieves the best performance over all the datasets, significantly outperforming all the baseline methods. Although review information is useful in recommendation, the performance can vary depending on how the review information is utilized. Our model integrates features extracted from reviews via an LSTM and—more importantly—applies attention networks to learn user and product representations instead of relying on a simple concatenation. Moreover, we propose an attention for utilizing text and rating information to capture the user's attention regarding different aspects of the product. This attention design allows each user preference to be modeled at a finer granularity. As the results show, this approach can lead to a better performance.

Finally, as the number of latent factors increases, AHUPA achieves the best performances on both datasets in Figure 2. Compared to SVD++, review-based methods are relatively stable, because they can already achieve relatively good performances even when the number of latent factors is small. This result demonstrates the benefits of considering review information in preference modeling. In addition, AHUPA outperforms NARRE on all the latent factors, which shows the benefits of capturing the attention weights for each user-product pair in the rating encoder while considering review helpfulness.



(a) Video Games

(b) CDs and Vinyl

**Fig. 2.** Performance of all competitors w.r.t. the number of latent factors on two larger datasets.

### 5.4. Model Analysis: Effectiveness of Multilayer Attentive Learning

To examine the effectiveness of the multilayer attention in the proposed model, we compare three model variants by removing one type of attention each time to evaluate its contribution to the performance. The results are shown in Fig. 3.

As Fig. 3 shows, the rating attention mechanism effectively improves model performance, which validates our idea that users pay different attention to different products. This result also indicates that the proposed rating attention effectively captures the weight changes that represent users' attention to different products. Moreover, word-level attention is also useful. This may be because recognizing and highlighting the important words using the word-level attention network helps in learning more informative review representations. In addition, review-level attention is also useful in our AHUPA approach because different reviews are informative to different degrees for both the users and the product representations. Differentiating between informative and less informative reviews can help the model learn more accurate representations of users and items. Finally, combining all three levels of attention further improves the performance of the proposed approach, which validates the effectiveness of the attention mechanism in the proposed approach.

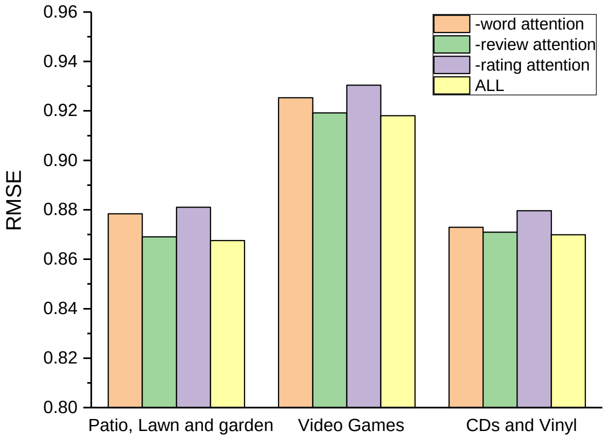


**Fig. 3.** Effectiveness of different attention layers

### 5.5. Model Analysis: Effectiveness of Attention Weights Based on Reviews and Ratings

To investigate the effects of the attention weight $\gamma^a_{u,p}$ and $\beta^a_{u,p}$ for each user-product pair in the rating encoder, we implemented two separate network models by removing $\gamma^a_{u,p}$ or $\beta^a_{u,p}$ from the attentive interaction. Fig. 4 shows the performance of a single network model with review-based ($\gamma^a_{u,p}$) or rating-based ($\beta^a_{u,p}$) attention information.

The results show that the review information is more effective than the rating information in enhancing the recommendation performance because reviews provide more valuable information than do ratings with regard to learning the changes in the users' attention weights for each user-product pair—especially in larger datasets. Moreover, the proposed model exhibits the highest performance, which validates the rationality of calculating attention weights based on reviews and ratings. The results also indicate that the proposed rating encoder can capture the attention weights of users pertaining to different aspects of each product.

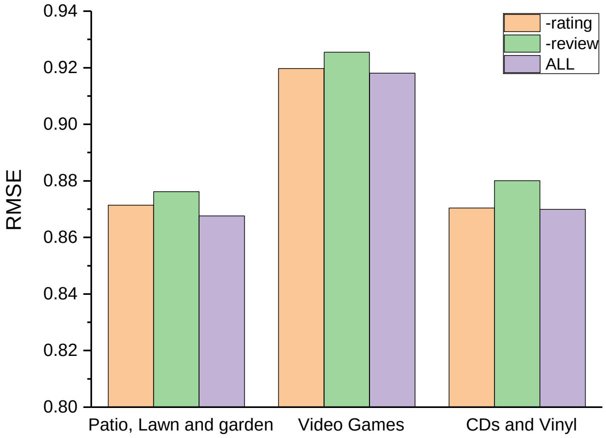
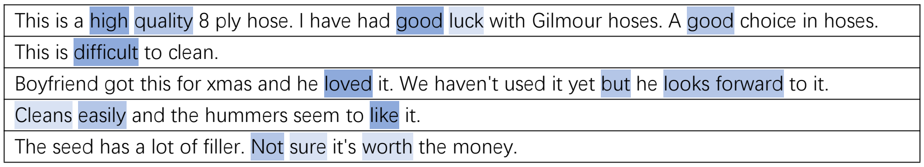


**Fig. 4.** Effectiveness of reviews and ratings

### 5.6. Visualization of Attention Weights

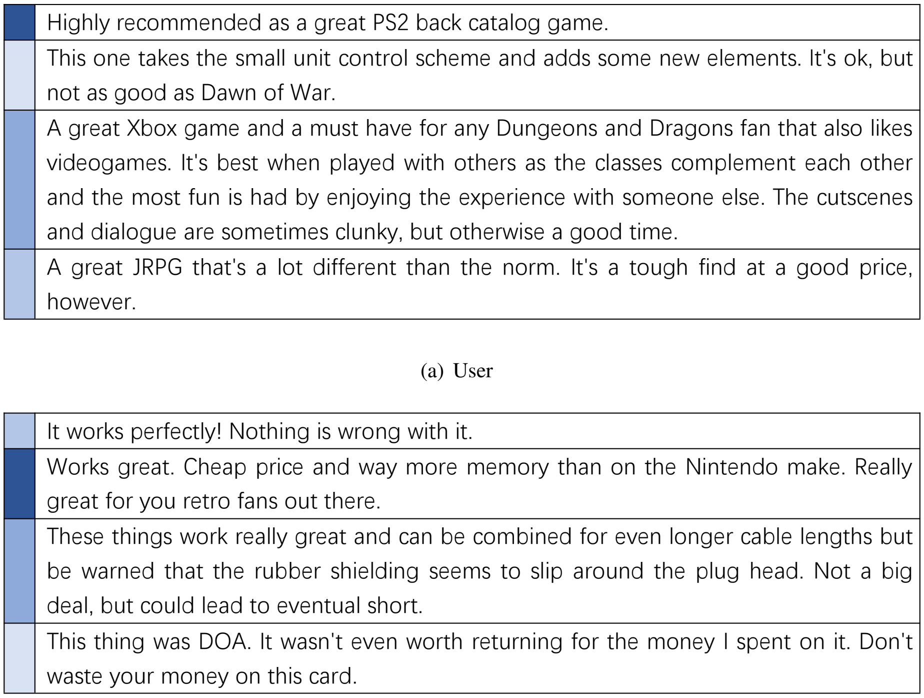
We conducted several case studies to verify the observations and demonstrate the effectiveness of the word- and review-level attention networks. First, we visualized the attention weights of the word-level attention networks, as shown in Fig. 5. According to Fig. 5, the word attention network effectively selects important words. For example, the words ”high” and ”quality” are assigned higher attention weights than are the words ”hoses” and ”clean”. This is because ”high” and ”quality” express more information than the other review words in terms of the user and product representations.

Second, the model effectively selects the important reviews by using the review level attention network. For example, the first review shown in Fig. 6(a) is assigned a high attention weight because it is highly informative for modeling the user preferences, while the second review shown in Fig. 6(a) is assigned a low attention weight because it contains limited information about the users. In addition, as shown in Fig. 6(b), the second review is assigned a higher attention weight than the fourth review because it is more valuable for modeling the product characteristics. These results validate that the proposed approach effectively evaluates the varying importances of words and reviews to highlight user preferences and product characteristics.



| |
|---|
| This is a high quality 8 ply hose. I have had good luck with Gilmour hoses. A good choice in hoses. |
| This is difficult to clean. |
| Boyfriend got this for xmas and he loved it. We haven't used it yet but he looks forward to it. |
| Cleans easily and the hummers seem to like it. |
| The seed has a lot of filler. Not sure it's worth the money. |

**Fig. 5.** Visualization of the word-level attention weights in randomly selected reviews from the Patio, Lawn and Garden dataset. A darker color corresponds to a higher attention weight



| | |
|---|---|
| ■ | Highly recommended as a great PS2 back catalog game. |
| ░ | This one takes the small unit control scheme and adds some new elements. It's ok, but not as good as Dawn of War. |
| ▓ | A great Xbox game and a must have for any Dungeons and Dragons fan that also likes videogames. It's best when played with others as the classes complement each other and the most fun is had by enjoying the experience with someone else. The cutscenes and dialogue are sometimes clunky, but otherwise a good time. |
| ▒ | A great JRPG that's a lot different than the norm. It's a tough find at a good price, however. |

(a) User

| | |
|---|---|
| ▒ | It works perfectly! Nothing is wrong with it. |
| ■ | Works great. Cheap price and way more memory than on the Nintendo make. Really great for you retro fans out there. |
| ▓ | These things work really great and can be combined for even longer cable lengths but be warned that the rubber shielding seems to slip around the plug head. Not a big deal, but could lead to eventual short. |
| ░ | This thing was DOA. It wasn't even worth returning for the money I spent on it. Don't waste your money on this card. |

(b) Product

**Fig. 6.** Visualization of review attention weights for a randomly selected user and product in the CDs and Vinyl dataset. A darker color corresponds to a higher attention weight

## 6. Conclusions

This paper proposes a novel multiaspect awareness model based on hierarchical user-product attention to realize rating prediction. We use a review encoder to learn the review representations from words and a user/product encoder to learn the user/product representations from the reviews. In addition, a rating encoder is introduced in AHUPA that utilizes the features extracted from reviews and ratings; this encoder can clarify the attention that a user pays to each aspect of the targeted product. Specifically, this attention design fully reflects the changes in the user's attention for the same aspect of different products. Experiments performed on three real-world datasets from Amazon demonstrate that the proposed approach effectively improves the recommended performance and outperforms many baseline methods.

In the future, we aim to more extensively examine user-product attention interactions to explore the different product aspects that interest users and develop improved embedded learning methods for extracting user and product characteristics by adopting better language models. Moreover, we aim to consider a larger amount of ancillary information, such as labels and attributes [7], that can better express the user interest and product characteristics.

**Acknowledgments.** This work is supported by the National Nature Science Foundation of China (No. 61972357, No. 61672337).

## References

1. Ahmed, K., Keskar, N.S., Socher, R.: Weighted transformer network for machine translation. ArXiv abs/1711.02132 (2018)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003)
4. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, Á.: Recommender systems survey. Knowl. Based Syst. 46, 109–132 (2013)
5. Catherine, R., Cohen, W.W.: Transnets: Learning to transform for recommendation. CoRR abs/1704.02298 (2017), http://arxiv.org/abs/1704.02298
6. Chen, C., Zhang, M., Liu, Y., Ma, S.: Neural attentional rating regression with review-level explanations. In: WWW (2018)
7. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G.S., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., Shah, H.: Wide deep learning for recommender systems. In: DLRS 2016 (2016)
8. Cheng, Z., Ding, Y., He, X., Zhu, L., Song, X., Kankanhalli, M.S.: $\mathbf{A^3NCF}$: An adaptive aspect attention model for rating prediction. In: IJCAI (2018)
9. Cheng, Z., Ding, Y., Zhu, L., Kankanhalli, M.S.: Aspect-aware latent factor model: Rating prediction with ratings and reviews. ArXiv abs/1802.07938 (2018)
10. Gao, H., Liu, C., Li, Y., Yang, X.: V2vr: Reliable hybrid-network-oriented v2v data transmission and routing considering rsus and connectivity probability. IEEE Transactions on Intelligent Transportation Systems pp. 1–14 (2020)
11. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang, X.: Context-aware qos prediction with neural collaborative filtering for internet-of-things services. IEEE Internet of Things Journal 7(5), 4532–4542 (2020)

12. Gao, H., Kuang, L., Yin, Y., Guo, B., Dou, K.: Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps. Mobile Networks and Applications pp. 1–16 (06 2020)
13. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: AISTATS (2011)
14. He, R., McAuley, J.J.: Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. ArXiv abs/1602.01585 (2016)
15. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. Proceedings of the 26th International Conference on World Wide Web (2017)
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9, 1735–1780 (1997)
17. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR abs/1412.6980 (2014)
18. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD (2008)
19. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42 (2009)
20. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. ArXiv abs/1405.4053 (2014)
21. Ling, G., Lyu, M.R., King, I.: Ratings meet reviews, a combined approach to recommend. In: RecSys '14 (2014)
22. Ma, X., Gao, H., Xu, H., Bian, M.: An iot-based task scheduling optimization scheme considering the deadline and cost-aware scientific workflow for cloud computing. EURASIP Journal on Wireless Communications and Networking 2019 (12 2019)
23. McAuley, J.J., Leskovec, J.: Hidden factors and hidden topics: understanding rating dimensions with review text. In: RecSys '13 (2013)
24. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: AAAI/IAAI (2002)
25. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS (2013)
26. Pazzani, M.J.: A framework for collaborative, content-based and demographic filtering. ARTIFICIAL INTELLIGENCE REVIEW 13, 393–408 (1999)
27. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: NIPS (2007)
28. 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. WWW01, Association for Computing Machinery, New York, NY, USA (2001), https://doi.org/10.1145/371920.372071
29. Seemann, P.: Matrix factorization techniques for recommender systems (2014)
30. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1929–1958 (2014)
31. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: End-to-end memory networks. In: NIPS (2015)
32. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: KDD (2011)
33. Wang, H., Zhang, F., Xie, X., Guo, M.: Dkn: Deep knowledge-aware network for news recommendation. In: WWW (2018)
34. Wang, X., He, X., Feng, F., Nie, L., Chua, T.S.: Tem: Tree-enhanced embedding model for explainable recommendation. In: WWW (2018)
35. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A.C., Salakhutdinov, R., Zemel, R.S., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: ICML (2015)
36. Yang, X., Zhou, S., Cao, M.: An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: The product-attribute perspective from user reviews. Mobile Networks and Applications (04 2019)

37. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., Hovy, E.H.: Hierarchical attention networks for document classification. In: HLT-NAACL (2016)
38. Zhang, Q., Wang, L., Meng, X., Xu, K., Hu, J.: A generic framework for learning explicit and implicit user-item couplings in recommendation. IEEE Access 7, 123944–123958 (2019)
39. Zhang, Y., Ai, Q., Chen, X., Croft, W.B.: Joint representation learning for top-n recommendation with heterogeneous information sources. In: CIKM '17 (2017)
40. Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., Ma, S.: Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (2014)
41. Zheng, L., Noroozi, V., Yu, P.S.: Joint deep modeling of users and items using reviews for recommendation. CoRR abs/1701.04783 (2017)

**ZHONGQIN BI** received his Ph.D. degree in system analysis and integration from East China Normal University, China in 2009. He is currently assistant professor in College of Computer Science and Technology, Shanghai University of Electric Power. Dr. Bi has published more than 30 articles in refereed journals and conference proceedings. His main research interests are cloud computing, data process and quality control in smart grid.

**SHUMING DOU** is currently working toward a master's degree in the College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai, China. His research interests include recommendation system, data process, and natural language processing.

**ZHE LIU** received his Master degree in electrical engineering from North China Electric Power University, China in 2016. He is currently engineer in State Grid Shanghai Electric Power Research Institute. He has published nearly 20 articles in refereed journals and conference proceedings. His main research interests are energy Internet and power quality.

**YONGBIN LI** received his MA. Sc in Computer Application Technology from Shanghai Jiaotong University, China in 2008. He is currently assistant professor in College of Computer Science and Technology, Shanghai University of Electric Power. MA. Li has published more than 10 articles in refereed journals and conference proceedings. His main research interests are data process and software engineering and he has won the third prize of Anhui provincial science and technology progress.

*Received: September 25, 2019; Accepted: July 1, 2020.*

Computer Science and Information Systems 17(3):867–890 https://doi.org/10.2298/CSIS190905025C

# A Study on the Development of a Light Scattering Particulate Matter Sensor and Monitoring System*

Jun-hee Choi[1] and Hyun-Sug Cho[2]

[1] Department of Disaster Prevention, Graduate School, Daejeon University
62, Daehak-ro, Dong-gu, Daejeon, Korea
junhee3020@hanmail.net
[2] Department of Fire & Disaster Prevention, Daejeon University
62, Daehak-ro, Dong-gu, Daejeon, Korea
chojo@dju.kr

**Abstract.** The gravimetric method, which is mainly used among particulate matter (PM) measurement methods, includes the disadvantages that it cannot measure PM in real time and it requires expensive equipment. To overcome these disadvantages, we have developed a light scattering type PM sensor that can be manufactured at low cost and can measure PM in real time. We have built a big data system that can systematically store and analyze the data collected through the developed sensor, as well as an environment where PM states can be monitored mobile in real time using such data. In addition, additional studies were conducted to analyze and correct the collected big data to overcome the problem of low accuracy, which is a disadvantage of the light scattering type PM sensor. We used a linear correction method and proceeded to adopt the most suitable value based on error and accuracy.

**Keywords:** Sensor Development, Fine Particles, Mobile Monitoring, Big Data, Light Scattering.

## 1. Introduction

As studies on the harmfulness of PM to the human body progress, PM has attracted great public interest, and as the times during which PM concentrations are high have increased, PM has become a serious social problem. In general, PM refers to particulate matter with particle sizes not larger than 10 $\mu m$, and occurs due to natural factors and artificial factors. In modern times, the ratio of PM generated due to artificial factors is high, and mainly those substances that have not been completely burned during the process of combustion of fossil fuels pollute the atmosphere. Fine particulate matter (FPM) is dust consisting of particles smaller than those of PM, and means fine particles not larger than PM 2.5 $\mu m$. As with PM, FPM is also produced by artificial factors, and because the particle sizes are smaller than

* Jun-hee Choi, Hyun-Sug Cho, "A Study on the Improvement of the Accuracy of Low-Cost Light Scattering Type Particulate Matter Sensors," Asia-pacific Society of Convergent Research Interchange, 2019.

those of PM, it may not be filtered out in the respiratory organs and can have fatal effects on the human body[1-2].

In order to solve the problems of PM, various efforts are being made both domestically and abroad [3-5], and studies regarding the measurement of PM and FPM are being conducted. PM measurement methods are divided into concentration measurement and size classification. In South Korea, PM is measured with concentration measurement methods. The concentration measurement methods comprise the gravimetric method, which collects PM for a certain time and measures the weight, and a method where *β*-rays are emitted to PM in the atmosphere and the *β*-rays absorbed by the PM are measured to determine the concentration of PM. These two methods are currently used interchangeably in South Korea, and show the highest accuracy among the measurement methods, but they require expensive equipment and provide data once an hour because they require preliminary work for measurement such as collecting and drying PM. To overcome these problem, light scattering type PM measurement methods have been studied[6]. The light scattering type methods use light for measurement and can measure the amount of PM in the atmosphere in real time. In addition, they have the advantage of enabling mass production at low prices since the light emitting part and the light receiving part play the largest roles among the components of the sensor. However, they have a vulnerable point in that the accuracy is relatively low because there is unit conversion two times with the measurement method. Due to this problem, PM data measured using light scattering type sensors have not been officially recognized until recently. However, the Ministry of Environment provided guidelines for light scattering type PM sensors in October 2018. The grades of data were divided according to the accuracy of measurements using PM sensors, and the data were divided with usable ranges, such as data for official measurement and those for research[7]. Undergoing correction work to meet such standards was recommended, and data with low rates of error based on the gravimetric method can be used as officially measured data. Correcting errors or increasing accuracy are applied in various ways in the big data field[8-17].

In this study, we developed a light scattering type PM sensor and built a system that stores data received from the sensor in a big data server. Based on these data, we established a mobile service that can monitor PM situations and conducted a study to correct the accuracy of the PM sensor[18]. we aimed to solve the problem of low accuracy in low-cost light scattering PM sensors.

## 2. Related Works

### 2.1 *β*-ray absorption method

The *β*-ray absorption method measures the amount of *β*-rays absorbed in the filter paper (filter) used to collect PM and obtains the concentration of PM from the measured value using the property of *β*-rays that are absorbed more when they pass through substances with larger masses[19]. The $\beta$-ray absorber removes particles not smaller than 10 $\mu m$ because a separator is installed in its air inhaling unit, and inhales air at a set flow rate to collect dust not larger than 10 $\mu m$ on the filter paper to measure the mass concentration of PM10. The PM10 separator is built in a certain structure that can separate particles by size using inertia and can be a cyclone type or an impactor type according to the shape of the particle size separation structure. If installed with an additional separator, the $\beta$-ray absorber can also measure the mass concentration of PM2.5 by collecting particles not larger than 2.5 $\mu m$. In the case of PM2.5, particles are separately collected by size through a primary separator (removing particles not smaller than 10 $\mu m$) and a secondary separator (removing particles not smaller than 2.5 $\mu m$) using the impactor method. The minimum detection limits of the measured mass concentrations of PM2.5 and PM10 are known to be 5 and 10 $\mu g/m^3$ or less, respectively, and the ranges of measured concentrations are 0~1,000 $\mu g/m^3$ in the case of PM2.5 and 0~1,000, 0~5,000, 0~10,000 $\mu g/m^3$, etc. in the case of PM10.

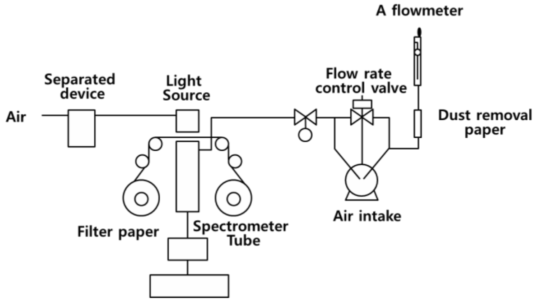


**Fig. 1.** $\beta$-ray measurement method operation principle

The $\beta$-ray absorption method measures PM concentrations by calculating the $\beta$-rays absorbed and dissipated when the $\beta$-rays irradiated from the light source that emits $\beta$-rays pass through the dust collected on the filter paper following Equations (1) and (2).

$$I = I_0 \times \exp(-\mu X) \qquad \textbf{(1)}$$

where I refers to the intensity of the $\beta$-rays that penetrated through the dust collected on the filter paper and $I_0$ refers to the intensity of the $\beta$-rays that penetrated through the blank filter paper. That is, $I_0$ means the intensity of the $\beta$-rays that penetrated through the filter paper on which no dust has been collected. $\mu$ is the $\beta$-rays mass absorption extinction coefficient ($cm^3/\mu g$) by PM, and can be said to be constant when the component and particle size distribution are constant. $X$ refers to the mass ($\mu g/cm^3$) of the dust collected per unit area. The $\beta$-ray intensity calculated through Equation (1) can be applied to Equation (2) to calculate the final PM concentration.

$$\mathrm{C}(\mu g/m^3) = \frac{S}{\mu \cdot Q \cdot \Delta t} \ln\left(\frac{I}{I_0}\right) \qquad \textbf{(2)}$$

where C refers to PM concentration, S refers to the area ($m^3$) of the filter paper on which PM was collected, V refers to amount of air inhaled ($m^3$), and $\Delta t$ refers to the sampling time (min). The reason for using $\beta$-rays, which are a sort of radiation, as a light source is that they have a property where they further lose their intensity when they pass through substances with larger masses. In addition, $\beta$-rays are safe because they cannot pass through metal plates or plastics, and enable comparison using initial filter paper as a background concentration because they pass through the collecting filter paper and reach the $\beta$-ray sensing unit. On the other hand, particle separators use inertial force, and even though their initial error range is very low, their separation capacity cannot but vary according to the weather conditions when foreign matter in the atmosphere adheres to them or the PM is highly viscous. Therefore, the principal limitation of a separator using a cyclone or impactor cannot but exist.

## 2.2 Gravimetric method

The gravimetric method collects samples for 24 hours and measures the mass of PM smaller than 2.5 $\mu m$ among the substances collected on the filter paper. The light scattering method measures the amount of scattered light using the principle that when light is cast on an object, the light collides with the object and is scattered in various directions and the concentration of PM is obtained from the measured value.

The gravimetric method is one of the methods of measuring the mass concentration of PM10 or PM2.5 in the atmospheric environment. It collects PM samples in the atmosphere using a sampler, and calculates the weight difference of the filter before and after the collection of PM samples as the mass concentration of the PM. The collected samples have the advantage of enabling physical and chemical analysis later, but substances such as sulfur dioxide ($SO_2$) or nitric acid that undergo chemical reactions to be oxidized into sulphate or nitrate have disadvantages of leading to the indication of excessive mass concentrations because solid salts are generated or resulting in errors in measurement results due to a reduction in the mass concentration as a result of the dissociation process. The limit of detection of the measured mass concentration should be within 5 $\mu g/m^3$ when the range of the measured mass concentrations is 80 $\mu g/m^3$ or lower and within 7% of the measured mass concentrations when the range exceeds 80 $\mu g/m^3$ on the basis of aerodynamic diameter in the case of PM10. The gravimetric method's limit of quantification of PM2.5 is 3 $\mu g/m^3$. Figure 2 shows a block diagram of the sampler used in the gravimetric method for PM2.5. First filter the PM10 samples by inhaling air from the atmosphere, and then measure the PM2.5 samples.

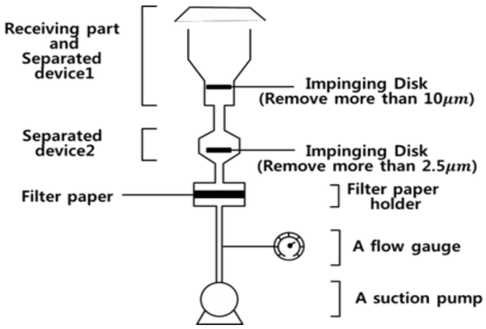

Receiving part and Separated device1
Impinging Disk (Remove more than 10$\mu m$)
Separated device2
Impinging Disk (Remove more than 2.5$\mu m$)
Filter paper
Filter paper holder
A flow gauge
A suction pump

**Fig. 2.** Operation principle of the gravimetric method

The collected samples have the advantage of enabling physical and chemical analysis later, but in the case of substances such as sulfur dioxide ($SO_2$) or nitric acid that undergo chemical reactions to be oxidized into sulphate or nitrate, the PM concentration measured is indicated in units of $\mu g/m^3$, which represents the weight of PM in 1 $m^3$ ($\mu g$ means one millionth gram). The weight concentration is indicated in units of $\mu g/m^3$ and is calculated by the following equation (3).

$$\mathrm{PM} = \frac{W_f - W_i}{V_a} \tag{3}$$

Both PM10 and PM2.5 are calculated with the same method. $Wf$ refers to the weight of filter paper after sampling ($\mu g$), $Wi$ refers to the weight of filter paper before sampling ($\mu g$), and $Va$ refers to the total sampling volume ($m$3). Here, the flow rate is corrected using the standard temperature and air pressure (0 °C, 1 atm) and is calculated as shown in equation (4).

$$V_a = Q \times \frac{293}{273+T} \times \frac{P}{760mmHg} \times t \tag{4}$$

Recently, various PM measuring devices have been developed using the light scattering measurement method because this method has various advantages such as the downsizing of measuring devices, real-time measurement, and dramatic reductions in the cost of device manufacturing. $T$ and $P$ represent the temperature and pressure at the time of measurement, respectively, and t refers to the total time of sampling. Particles not smaller than 10 $\mu m$ in the air sucked into the device by the suction pump collide with the impactor at the inlet due to inertial force and are separated from the main air stream. Thereafter, the PM particles that passed through the impactor are collected on the filter paper, and the weight is measured later to indicate the weight concentration of the PM particles in comparison with the air volume. The filter paper used in this case is nitrocellulose membrane filter paper, quartz fiber, or a Teflon material with an initial collection rate of at least 99% for 0.3 $\mu m$ particles, and the effect of moisture in the air should be minimized. The gravimetric method is divided into high-volume air samplers and low volume air samplers according to the volume of air collected. In the case of high-volume samplers, the suction flow should be about 2 $m^3$/min at no load and the sampler should be capable of continuous measurement for 24 h. The low volume air sampler can measure in a range of 10~30 L/min.

## 2.3 Light scattering measurement method

The light scattering method obtains the quantity of particulate matter using the principle that when light is cast on particulate matter (PM) suspended in the atmosphere, the light is scattered by the particles, and that when light is cast on particulate matter with the same physical properties, the amounts of scattered light are proportional to the mass concentrations of the particulate matter[6].

In order to understand climate change caused by the scattering effects of PM in the air, experiments have been carried out worldwide using a device called a Nephelometer. The three-wavelength scatterometer Nephelometer, which is the one most commonly used in the field, yields total scattering coefficients and back scattering coefficients for three wavelengths (λ) of 450, 550, and 700 nm. Photomultiplier tubes (PMT), which are the main element of the measuring unit, measure the quantities of scattered particles (number of photons) coming in through the scattering unit for three wavelengths of blue (450 nm), green (550 nm), and red (700 nm), respectively.

The spectrometer inhales samples in the air together with the air in the atmosphere at a constant flow rate and irradiates the laser to the particles passing through the measuring space inside the monitor. Then the laser beam is scattered by the particulate matter in the air flow. The light scattered this way is collected by the light concentrator and sent to the photodetector. The photodetector generates electrical signals in proportion to the amount of light concentrated, and the generated electrical signals are measured in pulses. The heights and numbers of the pulses can be converted into the diameters and numbers of particles for measurement and the outcome can be divided into 32 channels in a range from 0.23 to 32 m depending on the specification of the monitor. Let us take a closer look at the principle of the spectrometer. Both the hardware and software required to make a particle size measurement have gone through many revisions to improve accuracy, precision, reliability, and ease of use. At the very heart of the laser diffraction technique is the relationship between light and surfaces (which can be freely interchanged with "particle" for our purposes). When light strikes a surface, it is either diffracted, reflected, refracted, absorbed or reradiated.

$$\mathrm{I} = \frac{I_0 F(\theta,\phi)}{k^2 r^2} \qquad (5)$$

$I_0$: Intensity of incident light
I: Intensity of scattered light
θ: Scattering angle
∅: Azimuthal angle
k: Wave number
r: Distance from scatter to detector

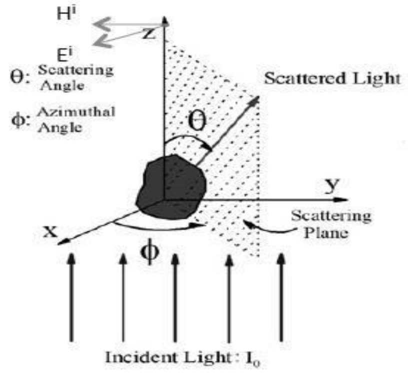
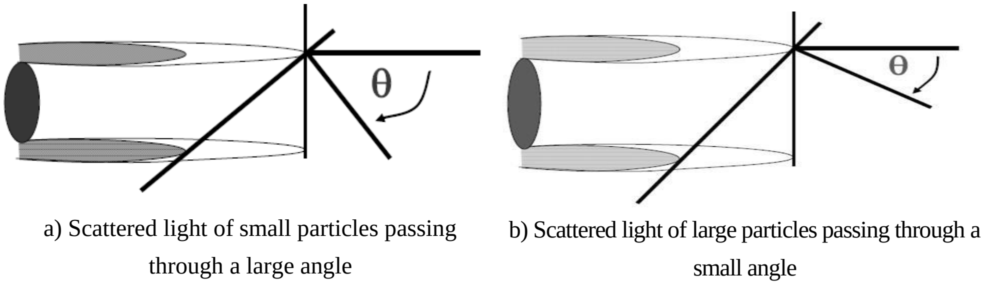


**Fig. 3.** Coordinate geometry for Rayleigh and Mie scattering[20]

For particles larger than a certain size, the vast majority of light is scattered by diffraction. The scattered light is at a relatively high intensity and low angle for these larger particles[21]. The "certain size" is determined as a multiple of the wavelength of light used for the measurement and is typically approximated at 20 microns. Particles larger than this size communicate useful size information through diffraction and not refraction. This means the measurement will not benefit from the use of a refractive index to accurately interpret refracted light.

For particles smaller than 20 microns, refracted light becomes increasingly important to calculate an accurate particle size. The scattered light is at a relatively low intensity and wide angle for these smaller particles.



a) Scattered light of small particles passing through a large angle

b) Scattered light of large particles passing through a small angle

**Fig. 4.** Relationship between distribution width and angular light intensity scattering patterns

The use of a refractive index and the Mie scattering theory directly affects accuracy in this size range. The basic workflow of a laser diffraction particle size analysis breaks down into two parts:

- Measure scattered light angle and intensity
- Transform that scattering data into a particle size distribution

Measurement quality refers to the analyzer itself: the quality of components, engineering refinement, and a fundamental design that reflects basic principles.

## 2.4 Decimation-in-Frequency (DIF) Radix-2 FFT Algorithm

The section considers radix-2 decimation in the frequency FFT algorithm and the Radix-4 Complex FFT algorithm that is applied to the developed measurement system[22].

### 2.4.1. Decimation-in-Frequency FFT Algorithm

In the context of fast Fourier transform algorithms, a butterfly is a portion of the computation that combines the results of smaller discrete Fourier transforms (DFTs) into a larger DFT, or vice versa (breaking a larger DFT up into sub-transforms). The name "butterfly" comes from the shape of the data-flow diagram in the radix-2 case. Most commonly, the term "butterfly" appears in the context of the Cooley–Tukey FFT algorithm, which recursively breaks down a DFT of composite size n = rm into r smaller transforms of size m where r is the "radix" of the transform.

Let N samples of the input signal, S(n), n=0, …N-1 be, at the same time N is the whole degree of two $N=2^L$. In the decimation-in-frequency algorithm the input S(n), n=0, …N-1, is halved, i.e. $S_0(m)=S(m)$ and $S_1(m)=S(m+N/2)$, m=0,1, …, N/2-1. Spectral samples S(2 · p), p=0, …N/2-1, with even indexes are N/2-point DFT of the signal $X_1(m)=S(m) +S(m+N/2)$. Similarly consider the signal S(n) for spectral samples S(2 · p+1), p=0,…N/2-1, with odd indexes.

Thus, the divide procedure includes calculating signals of half duration $X_0(m)$ and $X_1(m)$, m=0,…N/2-1. Then it is possible to replace N-points DFT with two N/2-points DFT of signals $X_0(m)$ and $X_1(m)$. The procedure for calculating signals of half duration can be presented in the form of a butterfly diagram as in Figure 5.

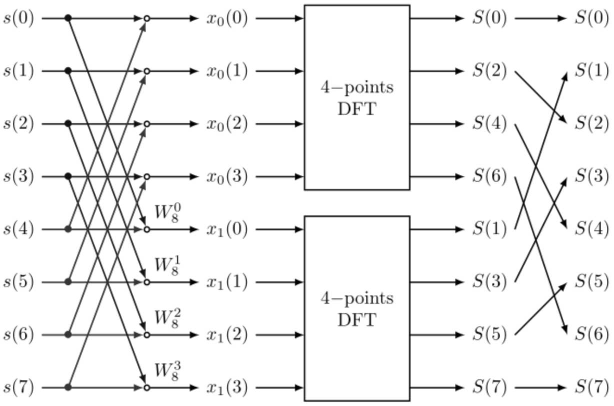


**Fig. 5.** Diagram of Decimation-in-Frequency FFT Algorithm for N=8

A decimation-in-time radix-2 FFT breaks a length-N DFT into two length-N/2 DFTs followed by a combining stage consisting of many butterfly operations.

### 2.4.2. Radix-4 Complex FFT Algorithm

This paper used Radix-4 Complex FFT Functions in the firmware algorithm for transferring from the electronic level of signal to frequency level of signal.

Complex fast Fourier transform (CFFT) and complex inverse fast Fourier transform (CIFFT) is an efficient algorithm to compute discrete Fourier transform (DFT) and inverse discrete Fourier transform (IDFT). The computational complexity of CFFT is reduced dramatically when compared to DFT. This set of functions implements CFFT/CIFFT for Q15, Q31, and floating-point data types. The functions operate on an in-place buffer that uses the same buffer for input and output. Complex input is stored in the input buffer in an interleaved fashion. The functions operate on blocks of input and output data and each call to the function processes 2*fft Len samples through the transform. pSrc points to In-place arrays containing 2*fft Len values.

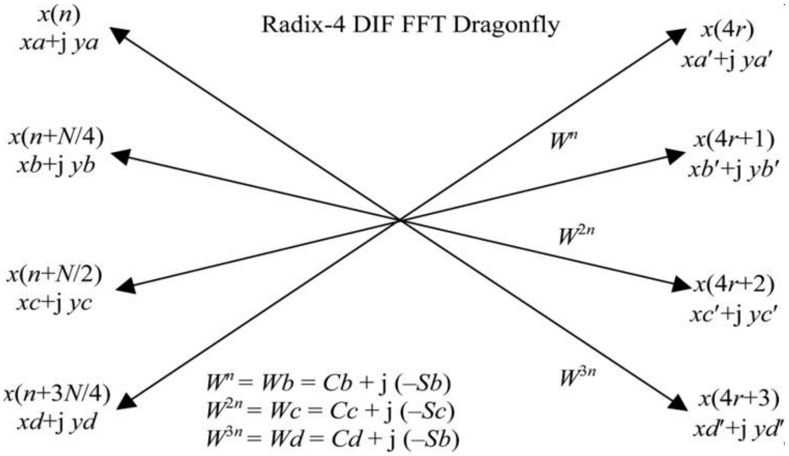


**Fig. 6.** Radix-4 Decimation-in Frequency Complex FFT

**Algorithm: Complex Fast Fourier Transform**
**Input real and imaginary data:**

x(n) = xa + j * ya
x(n+N/4 ) = xb + j * yb
x(n+N/2 ) = xc + j * yc
x(n+3N/4) = xd + j * yd

**where N is the length of FFT Output real and imaginary data:**

x(4r) = xa'+ j * ya'
x(4r+1) = xb'+ j * yb'
x(4r+2) = xc'+ j * yc'
x(4r+3) = xd'+ j * yd'

**Twiddle factors for radix-4 FFT:**

$W^n$ = Cb + j * (- Sb)
$W^{2n}$ = Cc + j * (- Sc)
$W^{3n}$ = Cd + j * (- Sb)

Different FFT algorithms provide different benefits, but there is always a trade-off between computation speed and FPGA area used. Reducing an FFT’s computation time reduces hardware complexity. Different decomposition methods are available, such as Radix-2 (Radix-2 nodes are known as Butterfly nodes) and Radix-4 (Radix-4 nodes are known as Dragonfly nodes).

**DIF Radix-4 CFFT Process Code:**

```
for (i = 0; i < SAMPLES; i += 2) {
  /* Each 22us ~ 45kHz sample rate */
    HAL_ADC_Start_IT(&hadc1); Delay(21);
  /* Real part, must be between -1 and 1 */
    Input[(uint16_t)i] = (float32_t)(ADC_val -
    (float32_t)2048.0) / (float32_t)2048.0;
  /* Imaginary part */
    Input[(uint16_t)(i + 1)] = 0; }
  /* Initialize the CFFT/CIFFT module, intFlag = 0, doBitReverse = 1 */
    arm_cfft_radix4_init_f32(&S, FFT_SIZE, 0, 1);
  /* Process the data through the CFFT/CIFFT module */
    arm_cfft_radix4_f32(&S, Input);
  /* Process the data through the Complex Magnitudes Module for calculating
  the magnitude at each bin */
    arm_cmplx_mag_f32(Input, Output, FFT_SIZE);
  /* Calculates maxValue and returns corresponding value */
   arm_max_f32(Output, FFT_SIZE, &maxValue, &maxIndex);
```

## 3. PM sensor development

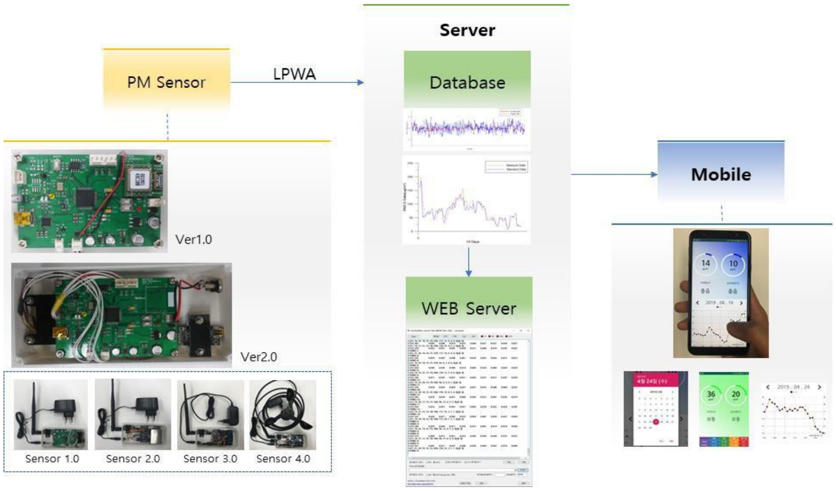


**Fig. 7.** Conceptual diagrams of the PM sensor, big data, and mobile monitoring service

The developed sensor is a complex sensor that measures all the factors that can affect PM measurement, and measures temperatures, humidity, and wind speeds based on the PM measuring function. The measured data are transmitted to the big data server through the long-range (LoRa) network of the low-power wide area (LPWA). The correction value of the PM sensor was calculated using the data stored in the server and was applied to the measurement sensor to improve accuracy, and finally, a service that can monitor the PM data on the desired date in the mobile terminal was built. An overall system block diagram is shown in Figure 7.

### 3.1 Sensor board development and modularization

The study developed a sensor board, which is the basis of the sensor. A block diagram is shown in Figure 8. The sensor board consists of a light emitting unit that shoots laser light at the sensing position, a light receiving unit that converts the received light into electrical signals and delivers the signals to the processor, and a heating unit that introduces air into the dust measurement sensor. Internally, a processing unit for converting electric signals into FFT to detect the PM that corresponds to the frequency band is included.

As for the process to detect PM, the light emitting unit shoots laser light at the air flowing into the sensor so that scattered light is generated by the PM included in the air. At this time, the fan introduces air into the sensor. The light receiving unit receives the light from the light emitting unit that is scattered by the PM, converts the light into electrical signals, amplifies the signals, and transmits the amplified signals to the processing unit. The processing unit identifies the frequency characteristics of the received electrical signals through FFT processing and measures those PM particles that correspond to individual frequency characteristics.

The LPWA module was connected to the rear of the sensor board to enable long-distance, low-power communication. Since the sensor board and LPWA module are included together in the case, they were connected by wire considering the issue of interference of wireless communication.

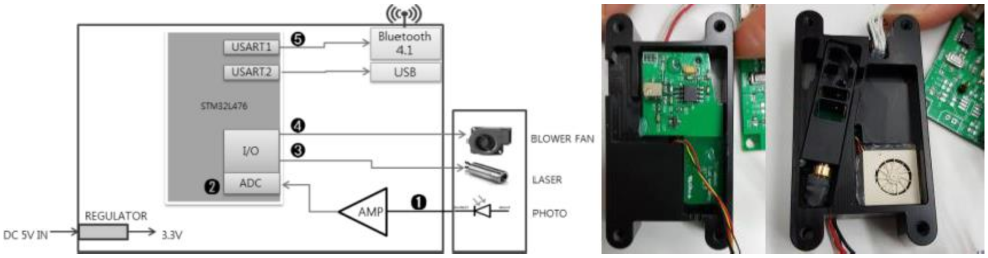


**Fig. 8.** The developed PM sensor board and sensor

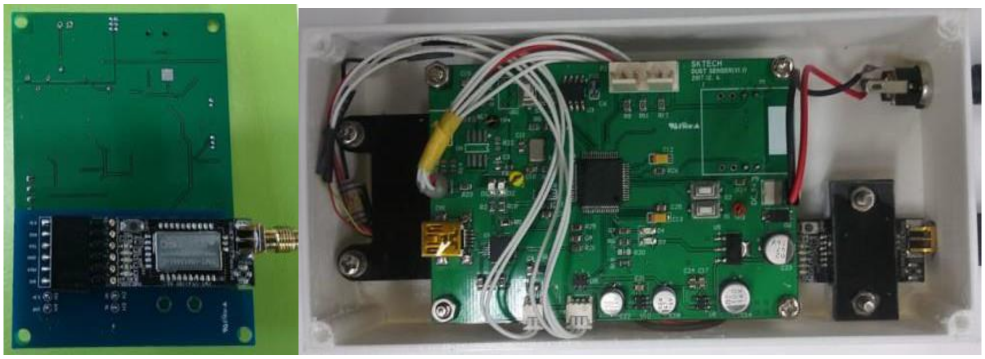


**Fig. 9.** The sensor board with an LPWA communication module applied, and the inside of the PM sensor module

## 3.2 Sensor housing

The exterior of the PM sensor was fabricated with a 3D printer. The exterior is a plastic case to contain sensor elements, cables, and the control board. the case is made for fabrication of the prototype in a size useful to mount Arduino and Raspberry Pi, and the sensor board was built in the case later.

Inside the sensor case, a PM sensor, a PM sensor board, an LPWA communication module, various cables were built in, and the exterior was fabricated to enable the connection of the antenna and the power supply.

## 3.3 Low power mode implementation

The PM sensor must be usable with a battery because it should be able to measure PM anywhere in real time. To realize the foregoing, a processor low power mode was implemented by reducing the permanent power with a view to securing hours of use not shorter than one month when general batteries are used. At the beginning, the low power mode was implemented so that the PM sensor would measure data every second and transmit the data immediately after measurement. However, since the power consumed for wireless communicating to store the data in the server was very large, a method of reducing the number of transmission times was chosen. The low power mode was implemented so that the PM sensor would identically operate every second, but the data would be transmitted every 10 minutes or only when requested by the server, and as a result, the power consumption decreased to a level of 20%.

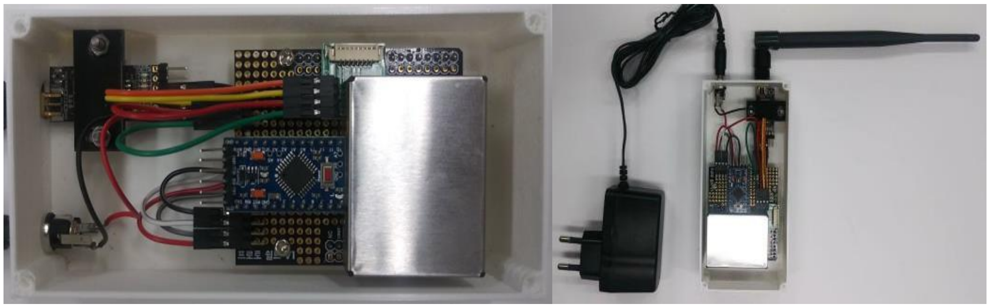


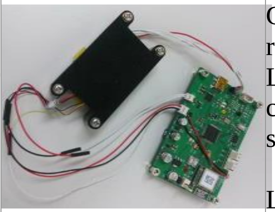
**Fig. 10.** PM sensor in which the low power mode was implemented.

## 3.4 Sensor advancement

In cases where PM is measured with light scattering type sensors, the measurement accuracy is affected by external environmental factors. To measure these variables and reflect them in the correction, the sensor was advanced by adding additional sensors that can measure various factors. Since additional power is necessary when additional sensors are attached, the advanced sensor is more suitable for installation types than for portable types. Additional sensor elements are included in Table 1. The composite sensors according to the combinations of sensor components are as follows.

- Sensor module A: Arduino UNO + additional dust sensor + temperature and humidity sensor + LPWA communication module, 9~12V power supply
- Sensor module B: Arduino Pro Mini + additional dust sensor + LPWA communication module, 5V power supply
- Sensor module C: Dust sensor self-developed in the first year + LPWA communication module, 5V power supply
- Sensor module D: Arduino UNO + wind speed sensor + atmospheric pressure/altitude sensor + communication module, 12V power supply

**Table 1.** Sensor components

| Division | Shape | Specification | Remark |
|---|---|---|---|
| Anemometer | | **Operating voltage:** 12V~ 24V<br>Analog Output-Use after ADC Conversion<br>**Power Consumption:**<br>Below 0.3W<br>**Resolution:** 0.1 m/s<br>**Effective measurement range:**<br>0 ~ 30m/s<br>**Operating Temperature:**<br>-40 ~ 80 ℃ | Mounted on wind speed / pressure sensor module |
| Barometer/ Altimeter | | **Operating voltage:** 3.3V<br>Barometer<br>Altitude measurement<br>TTL 232 Communications (fixed at 11520 bps)<br>set to 0.1 Hz~10 Hz | Mounted on wind speed / pressure sensor module |
| Dust sensor self-developed in the first year |  | **Operating voltage:** 5V<br>Combustion floating matter reaction<br>LPWA communication module combined light scattering type sensor applied Cortex-M4 core<br>LED on / off, Fan on / off functions | Specialized into combustibles sensor |
| Additional PM sensor | | **Operating voltage:** 5V<br>Particle Count Function Light scattering type<br>TTL-232 communication<br>Passive / Active mode switching | Mounted on PM and temperature/ humidity sensor module |
| LPWA Communications module | | **Operating voltage:** 3.3V<br>Low power broadband communication device<br>1:1 pairing mode<br>1:N mode support<br>Use 400 MHz band | Mount on all sensor modules |

## 4. Improvement of PM measurement accuracy of light scattering type sensors

To improve the accuracy of the sensor, a “data-based correction algorithm” was applied. PM standard data provided by Air Korea were used as the standard for accuracy and the data measured PM using the gravimetric method and $\beta$-rays. The PM concentration is indicated using units of $\mu g/m3$ and represents the weight of PM. In the experiment, data for 10 days in the Daejeon region in 2019 were used, and a data-based correction algorithm was applied using the standard data and measured data.

### 4.1 Dataset

Whereas the light scattering type PM sensor can measure PM in real time, the standard data are output every hour because the gravimetric method and $\beta$-rays are used. Therefore, the data collected using the light scattering type sensor were used in the correction work using the 1-hour average values.

**Table 2.** Data set used in the experiment

| Division | Daejeon standard data | Data measured by the sensor |
|---|---|---|
| **Period** | 10 days (240h) | 10 days (240h) |
| **Unit (Time)** | 1 Hour | 1 Second → 1 Hour(average) |
| **Unit (PM)** | 10 / 2.5 $\mu g/m^3$ | 10 / 2.5 / 1 $\mu g/m^3$ |
| **Method** | Mass Concentration (Manual) | Sensor (Auto) |

The dataset and test set were divided at a ratio of 8:2 to avoid overfitting in the data-based correction training results. The data for the first 8 days were used to carry out correction learning. Thereafter, the results were applied to the data for remaining 2 days of data and the results were checked.

### 4.2 Evaluation indicator

To quantify the difference between standard data and measured data, root mean square error (RMSE), which is a value that shows the difference between the two data, was used as an evaluation index. RMSE can be expressed as shown in Equation 6, where $\hat{s}_i$ is the data measured using the sensor, $d_i$ is the standard data, and $n$ is the total number of data.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}|\hat{s}_i - d_i|^2}{n}} \tag{6}$$

Accuracy means the ratio to which the data are identical to the four levels of PM provided by the Ministry of Environment. A value of 1 is given when the levels of the two data are identical, and 0 is given otherwise. Therefore, the data will have a value of 1 when all the levels are identical. The PM indicator is as shown in Table 3.

**Table 3.** PM indicator($\mu g/m^3$)

| | Good | Moderate | Bad | Very bad |
|---|---|---|---|---|
| **PM10** | ~30 | ~80 | ~150 | 151~ |
| **PM2.5** | ~15 | ~35 | ~75 | 76~ |

## 4.3 Experiment

In order to correct the data collected with the sensor, the slope value of the data was changed or parallel translated and corrected using the combination with which the RMSE becomes the smallest. The slope was changed from -10 to 10 in units of 0.1 and the parallel translation was calculated from -50 to 120 in the changed units of 1.

**Table 4.** Performance evaluation results (Test data)

| Data | | RMSE | Accuracy |
|---|---|---|---|
| Measure Data | PM10 | 26.3764 | 0.1666 |
| | PM2.5 | 12.8299 | 0.8333 |
| Revise Data (Slope) | PM10 | 18.0323 | 0.3750 |
| | PM2.5 | 9.1544 | 0.8958 |
| Revise Data (Shift) | PM10 | 9.5686 | 0.8750 |
| | PM2.5 | 8.7376 | 0.8333 |
| Revise Data (Slope + Shift) | PM10 | 9.5686 | 0.8750 |
| | PM2.5 | 10.0550 | 0.8541 |

The resulting values according to the performance evaluation indicator are shown in Table 4. It can be seen that the corrected data generally show higher performance values than the measured data. Although the particulate matter (PM10) showed high RMSE values and low accuracy when there was no correction, it could be identified that the performance was greatly improved by the correction work. The case of parallel translation showed the highest performance value, and given that there was no change even when the slope was applied, it could be predicted that the data were measured to be low in general. From the viewpoint of accuracy, the highest accuracy value was shown when the slope was applied. The FPM (PM2.5) also showed the highest RMSE performance when corrected by parallel translation, but from the viewpoint of accuracy, it showed the highest accuracy when corrected using slopes. In addition to slopes and parallel translation, various methods such as secondary correction equations and leveling-off were used, but it was identified that there was no big difference in terms of performance. With these results, the limits of mathematical correction were recognized, and it is expected that if the correlations of factors that affect PM are analyzed and the results are reflected, there will be additional performance improvement.

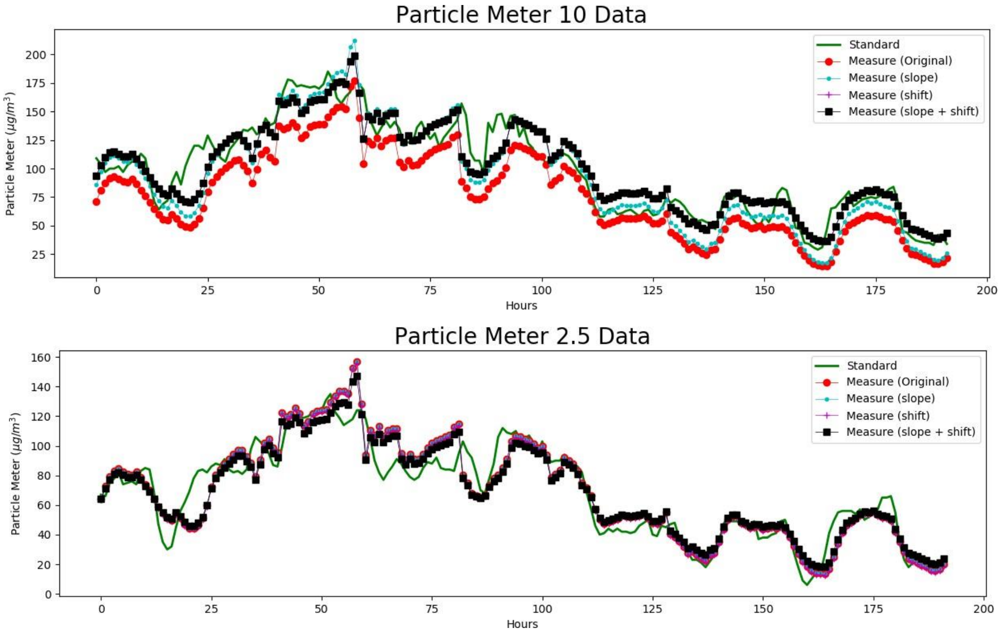


**Fig. 11.** PM(PM10) and FPM(PM2.5) data for 8 days

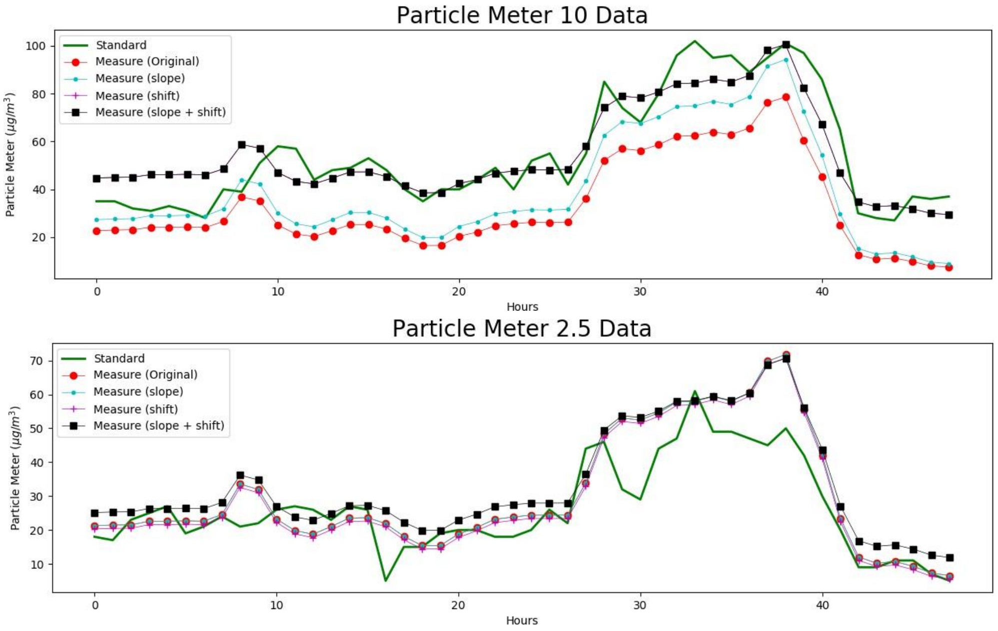


**Fig. 12.** PM(PM10) and FPM(PM2.5) data for 2 days

## 5. Mobile monitoring system

Since the data collected by the sensor must be transmitted to a server at a long distance, long- distance communication must be possible while the power is managed. Therefore, the LPWA communication-based LoRa method was applied to store the data in the server, which were then provided to the mobile application to build the PM monitoring system.

### 5.1 Low power technology (LoRa)

LoRa, which is one of the LPWA technologies, is a communication technology suitable for cases where low data transmission rates (<1Mb / s) that require low power consumption are required. LoRa's physical layer uses spread spectrum modulation (SSM) and also uses a way to encode the fundamental signal into a higher frequency sequence. This enables the fundamental signals to be intentionally spread over a wider bandwidth, reducing power consumption and improving tolerance to electromagnetic interference.

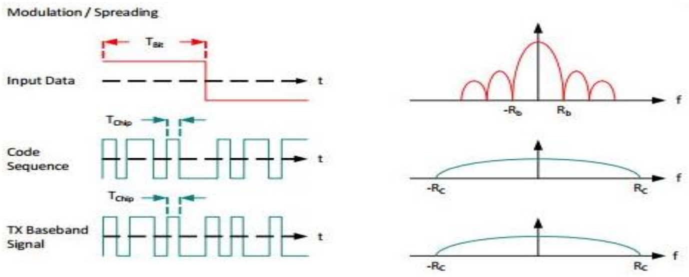


**Fig. 13.** Frequency characteristics according to signal lengths

In addition, SSM-based chirp spread spectrum (CCS) modulation is fundamentally sinusoidal signals that can be modulated into a wideband frequency that increases or decreases over time because it encodes data through chirp.

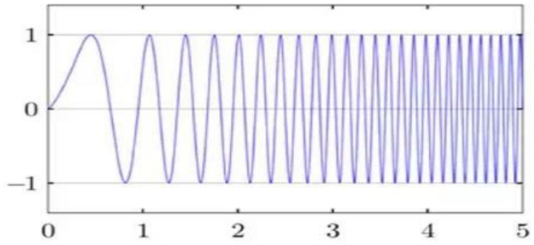


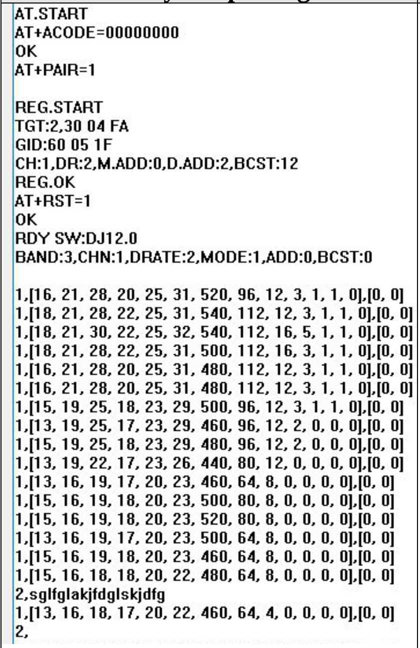
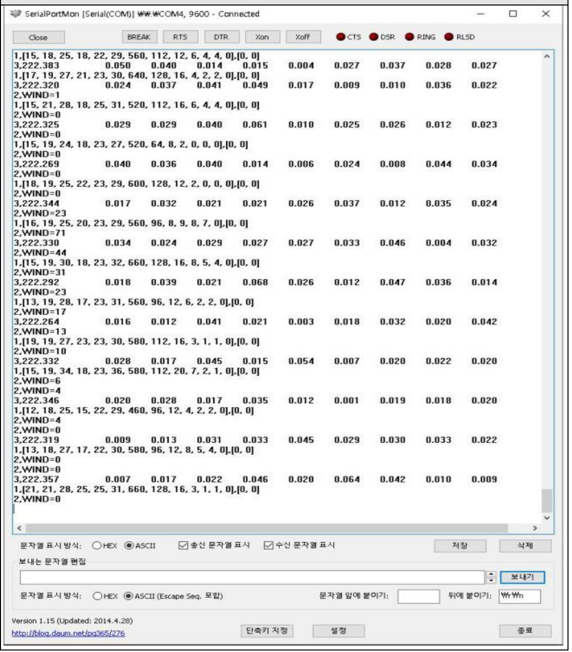
**Fig. 14.** Chirp spread spectrum (CSS) modulation

The network server is connected to multiple gateways through a secure TCP / IP connection and uses a way to eliminate duplicate messages and determine the gateway that should respond to end node messages. In addition, end-node data transmission rates are managed through the adaptive data transmission rate (ADR) method so that the network capacity can be maximized and the end-node battery life can be extended.

## 5.2 1: N star mode network of LPWA modules

Each PM sensor has an LPWA communication module that is connected to the big data server with 1: N star mode networks. By pairing, the band, channel and group ID of each transmission/receiving module are matched, address numbers are assigned sequentially, and the modules are implemented to carry out auto pairing when the power supply is reset later.

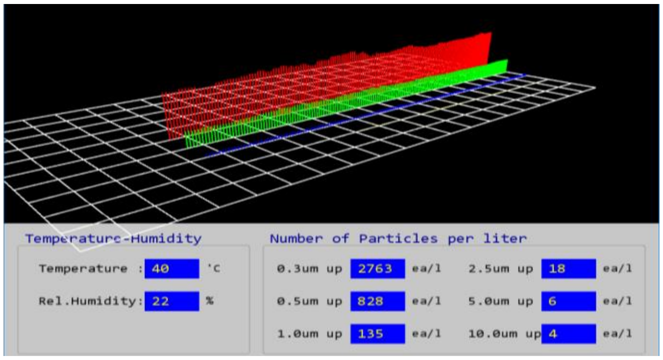
**Table 5.** Implementation of 1:N star mode networks

| Carry out pairing | Form 1:3 LPWA networks |
|---|---|
| <br>AT.START<br>AT+ACODE=00000000<br>OK<br>AT+PAIR=1<br><br>REG.START<br>TGT:2,30 04 FA<br>GID:60 05 1F<br>CH:1,DR:2,M.ADD:0,D.ADD:2,BCST:12<br>REG.OK<br>AT+RST=1<br>OK<br>RDY SW:DJ12.0<br>BAND:3,CHN:1,DRATE:2,MODE:1,ADD:0,BCST:0<br><br>1,[16, 21, 28, 20, 25, 31, 520, 96, 12, 3, 1, 1, 0],[0, 0]<br>1,[18, 21, 28, 22, 25, 31, 540, 112, 12, 3, 1, 1, 0],[0, 0]<br>1,[18, 21, 30, 22, 25, 32, 540, 112, 16, 5, 1, 1, 0],[0, 0]<br>1,[18, 21, 28, 22, 25, 31, 500, 112, 16, 3, 1, 1, 0],[0, 0]<br>1,[16, 21, 28, 20, 25, 31, 480, 112, 12, 3, 1, 1, 0],[0, 0]<br>1,[16, 21, 28, 20, 25, 31, 480, 112, 12, 3, 1, 1, 0],[0, 0]<br>1,[15, 19, 25, 18, 23, 29, 500, 96, 12, 3, 1, 1, 0],[0, 0]<br>1,[13, 19, 25, 17, 23, 29, 460, 96, 12, 2, 0, 0, 0],[0, 0]<br>1,[15, 19, 25, 18, 23, 29, 480, 96, 12, 2, 0, 0, 0],[0, 0]<br>1,[13, 19, 22, 17, 23, 26, 440, 80, 12, 0, 0, 0, 0],[0, 0]<br>1,[13, 16, 19, 17, 20, 23, 460, 64, 8, 0, 0, 0, 0],[0, 0]<br>1,[15, 16, 19, 18, 20, 23, 500, 80, 8, 0, 0, 0, 0],[0, 0]<br>1,[15, 16, 19, 18, 20, 23, 520, 80, 8, 0, 0, 0, 0],[0, 0]<br>1,[13, 16, 19, 17, 20, 23, 500, 64, 8, 0, 0, 0, 0],[0, 0]<br>1,[15, 16, 19, 18, 20, 23, 460, 64, 8, 0, 0, 0, 0],[0, 0]<br>1,[15, 16, 18, 18, 20, 22, 480, 64, 8, 0, 0, 0, 0],[0, 0]<br>2,sglfglakjfdglskjdfg<br>1,[13, 16, 18, 17, 20, 22, 460, 64, 4, 0, 0, 0, 0],[0, 0]<br>2, | <br>SerialPortMon [Serial(COM)] ₩₩.₩COM4, 9600 - Connected<br>1,[15, 18, 25, 18, 22, 29, 560, 112, 12, 6, 4, 4, 0],[0, 0]<br>3,222.383 0.050 0.040 0.014 0.015 0.004 0.027 0.037 0.028 0.027<br>1,[17, 19, 27, 21, 23, 30, 640, 128, 16, 4, 2, 2, 0],[0, 0]<br>3,222.320 0.024 0.037 0.041 0.049 0.017 0.009 0.010 0.036 0.022<br>2,WIND=1<br>1,[15, 21, 28, 18, 25, 31, 520, 112, 16, 6, 4, 4, 0],[0, 0]<br>2,WIND=0<br>3,222.325 0.029 0.029 0.040 0.061 0.010 0.025 0.026 0.012 0.023<br>2,WIND=0<br>1,[15, 19, 24, 18, 23, 27, 520, 64, 8, 2, 0, 0, 0],[0, 0]<br>2,WIND=0<br>3,222.269 0.040 0.036 0.040 0.014 0.006 0.024 0.008 0.044 0.034<br>2,WIND=0<br>1,[18, 19, 25, 22, 23, 29, 600, 128, 12, 2, 0, 0, 0],[0, 0]<br>2,WIND=0<br>3,222.344 0.017 0.032 0.021 0.021 0.026 0.037 0.012 0.035 0.024<br>2,WIND=23<br>1,[16, 19, 25, 20, 23, 29, 560, 96, 8, 9, 8, 7, 0],[0, 0]<br>2,WIND=71<br>3,222.330 0.034 0.024 0.029 0.027 0.027 0.033 0.046 0.004 0.032<br>2,WIND=44<br>1,[15, 19, 30, 18, 23, 32, 660, 128, 16, 8, 5, 4, 0],[0, 0]<br>2,WIND=31<br>3,222.292 0.018 0.039 0.021 0.068 0.026 0.012 0.047 0.036 0.014<br>2,WIND=23<br>1,[13, 19, 28, 17, 23, 31, 560, 96, 12, 6, 2, 2, 0],[0, 0]<br>2,WIND=17<br>3,222.264 0.016 0.012 0.041 0.021 0.003 0.018 0.032 0.020 0.042<br>2,WIND=13<br>1,[19, 19, 27, 23, 23, 30, 580, 112, 16, 3, 1, 1, 0],[0, 0]<br>2,WIND=10<br>3,222.332 0.028 0.017 0.045 0.015 0.054 0.007 0.020 0.022 0.020<br>1,[15, 19, 34, 18, 23, 36, 580, 112, 20, 7, 2, 1, 0],[0, 0]<br>2,WIND=6<br>2,WIND=4<br>3,222.346 0.020 0.028 0.017 0.035 0.012 0.001 0.019 0.018 0.020<br>1,[12, 18, 25, 15, 22, 29, 460, 96, 12, 4, 2, 2, 0],[0, 0]<br>2,WIND=4<br>2,WIND=0<br>3,222.319 0.009 0.013 0.031 0.033 0.045 0.029 0.030 0.033 0.022<br>1,[13, 18, 27, 17, 22, 30, 580, 96, 12, 8, 5, 4, 0],[0, 0]<br>2,WIND=0<br>2,WIND=0<br>3,222.357 0.007 0.017 0.022 0.046 0.020 0.064 0.042 0.010 0.009<br>1,[21, 21, 28, 25, 25, 31, 660, 128, 16, 3, 1, 1, 0],[0, 0]<br>2,WIND=0<br>문자열 표시 방식: ○HEX ●ASCII ☑송신 문자열 표시 ☑수신 문자열 표시<br>보내는 문자열 편집<br>문자열 표시 방식: ○HEX ●ASCII (Escape Seq. 포함) 문자열 앞에 붙이기: 뒤에 붙이기: ₩r₩n<br>Version 1.15 (Updated: 2014.4.28)<br>http://blog.daum.net/pg365/276 |

### 5.3 Big data server and monitoring software

The big data server stores PM /wind speed /atmospheric pressure /temperature and humidity data received through the LPWA module in the database and processes the statistical data and delivers the outcome to an Android app. Separate data visualization software was developed to analyze the data stored in the big data server, shown in Figure 15.

As for the process through which the PM data of the server are transmitted to the mobile app, when the PM sensor data monitoring request has been delivered to the HTTP protocol, the Apache web server accesses MySQL via JDBC, requests the query from the server, and transmits the collected PM data to the Android app as a web response.



Temperature-Humidity

Temperature : 40 'C

Rel.Humidity: 22 %

Number of Particles per liter

```
0.3um up 2763 ea/l   2.5um up  18 ea/l
0.5um up 828  ea/l   5.0um up  6  ea/l
1.0um up 135  ea/l   10.0um up 4  ea/l
```

**Fig. 15.** Real time data visualization

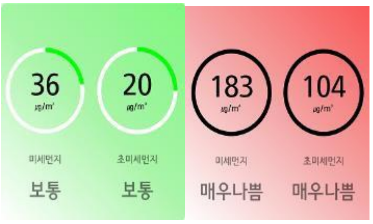
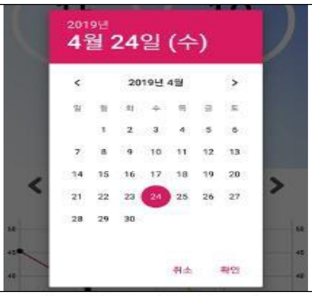
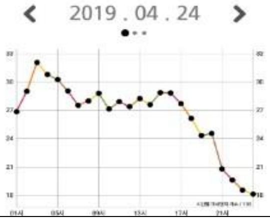
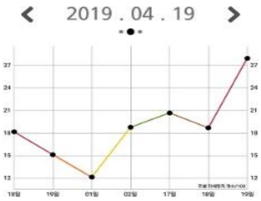
### 5.4 Development of mobile application

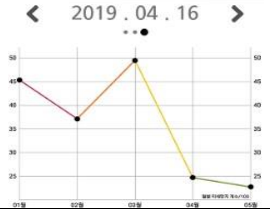
We developed an application that receives data by making a web connection in the data server, sending requests, and receiving web responses and outputting the data as a graph. The application was implemented to enable stable use by connecting and disconnecting the socket for each response when receiving data with a web connection and terminating the socket when the necessary data were transmitted.

The application outputs real-time PM and FPM concentration values and enables selecting desired dates to identify the PM concentration in the time zone. The PM and FPM concentration values are output to meet the four level standards according to the PM / FPM concentrations provided by the Ministry of Environment, and monthly, daily and hourly statistical data can also be output together and checked. This application works with Android 9.0 and above.

To ensure the quality of the software, it was registered. The PM10 and PM2.5 data to be displayed were correctly output by date and time, and the response time was within 1 second. The program is based on the Android operating system, so it is compatible with various mobile devices. However, in the process of importing the data, the loading time took several seconds for the massive amount of data. To prevent this, it was improved by efficiently loading the data. But access by a large number of users can lead to the same problems, and efficient allocation of resources is required to address these problems[23-24]. Research has also been conducted to select and analyze information that users need from big data[25-26].

**Table 6.** Description of the monitoring app user interface

| Division | Screen | Remark |
|---|---|---|
| Real-time PM status |   36<br>㎍/㎥<br>미세먼지<br>보통<br>20<br>㎍/㎥<br>초미세먼지<br>보통<br>183<br>㎍/㎥<br>미세먼지<br>매우나쁨<br>104<br>㎍/㎥<br>초미세먼지<br>매우나쁨 | - The current status can be checked with the air environment standard for PM (Framework Act on Environmental Policy)<br>- PM concentration standard 0-30 (good), 31-80 (moderate), 81-150 (bad), 151- (very bad)<br>- FPM concentration standard 0-15 (good), 16-50 (moderate), 51-100 (bad), 101- (very bad)<br>- Screen color conversion (background / circular) Good (blue / blue), moderate (green / green), Bad (orange / red), Very bad (red / black) |
| Change statistics check date |   2019년<br>4월 24일 (수)<br>2019년 4월<br>일 월 화 수 목 금 토<br>1 2 3 4 5 6<br>7 8 9 10 11 12 13<br>14 15 16 17 18 19 20<br>21 22 23 24 25 26 27<br>28 29 30<br>취소<br>확인 | - The calendar can be called up by tapping the date in the middle of the screen<br>- The date can be changed by day by clicking the left and right arrows in the middle of the screen<br>- The hourly, daily and monthly statistical data can be checked by selecting the desired date from the calendar. |
| Hourly data statistics |  | - Hourly PM is cumulatively displayed in the first tab on the bottom graph<br>- The cumulative PM count per hour is displayed in 1/100 units<br>- A screen swipe function enables moving to daily and monthly data statistics |
| Daily data statistics |  | - The second tab in the bottom graph cumulatively displays daily PM<br>- The cumulative daily PM counts are displayed in 1/100 units<br>- A screen swipe function enables moving to hourly and monthly data statistics |

| | | |
|---|---|---|
| Monthly data statistics |  2019 . 04 . 16<br>50<br>45<br>40<br>35<br>30<br>25<br>01월<br>02월<br>03월<br>04월<br>05월 | - The third tab in the bottom graph cumulatively displays monthly PM<br>- The cumulative monthly PM counts are displayed in 1/100 units.<br>- A screen swipe function enables moving to hourly and daily data statistics |

## 6. Conclusion and future works

Among the methods of measuring PM, the $\beta$-rays method and the gravimetric method are used as official data because they have high accuracy. However, the problem is that the equipment is expensive and measurement in real time is impossible. In order to solve this problem, a light scattering method for measuring PM has been proposed, but it has the disadvantage of low accuracy. However, it could be used for official data if it satisfied high accuracy. In general, hardware performance was improved during the manufacturing stage of the sensor, but there were no studies that performed the process of correcting the sensor based on data.

The measured values were linearly corrected to satisfy high accuracy of the light scattering sensor, and the best performance was selected based on the RMSE and accuracy. The correction result is approximately five times more accurate than before correction, and the RMSE is much improved. However, since the collection method of the mathematical model is based on PM data of the gravimetric method, there is a limit to performance improvement. Since the light scattering type sensor is greatly affected by external factors when measuring PM, factors that affect PM measurement and their correlations will be analyzed for better correction in future studies.

We developed a light scattering type PM measurement sensor that can be supplied at a low price and enables measurements in real time. A low power mode is implemented in the sensor. It is powered by commercial batteries for use in a mobile environment, and enters a low-power mode during standby to allow long-term use. In addition to the PM sensor, wind speed, barometric pressure, and altitude sensors are attached. An LPWA communication module was attached to the PM sensor and LPWA LoRa was used as the communication method to transmit to the server. It connects to a 1: N star mode network and collects several PM sensor data in one server.

The mobile monitoring system was developed to check real-time PM data and recorded data through a mobile application. The data measured in real time are transmitted to the server at certain intervals and stored there. The data are analyzed and applied with a sensor correction value. It provides a PM monitoring service using the data stored in the big data server, and users can access all data such as monthly, daily and desired date in the server. Since the PM sensor was developed and a system to monitor it was built, it is expected that the PM information of the area where the PM sensors are installed can be easily accessed.

Although the accuracy was improved by an error correction algorithm, it is necessary to analyze the cause of the error. It will further analyze the causes that affect the concentration of PM by using data such as moisture or wind speed in the atmosphere among the data collected through sensors.

**Acknowledgements.** This research was supported by the Daejeon University Research Grants (2016)

## References

1. Loomis, D.; Huang, W.; Chen, G. The International Agency for Research on Cancer (IARC) evaluation of the carcinogenicity of outdoor air pollution: Focus on China. Chin. J. Cancer 2014, 33, 189–196.
2. Ki-Hyun Kim, Ehsanul Kabir, Shamin Kabir, "A review on the human health impact of airborne particulate matter," Environment International, Vol. 74, pp. 136-143, Jan., 2015.
3. Kwasny, F.; Madl, P.; Hofmann, W. Correlation of air quality data to ultrafine particles (UFP) concentration and size distribution in ambient air. Atmosphere 2010, 1, 3–14.
4. Costa, M.A.M.; Carvalho, J.A.; Neto, T.G.S.; Anselmo, E.; Lima, B.A.; Kura, L.T.U.; Santos, J.C. Real-time sampling of particulate matter smaller than 2.5 μm from Amazon forest biomass combustion. Atmos. Environ. 2012, 54, 480–489.
5. França, D.D.A.; Longo, K.M.; Neto, T.G.S.; Santos, J.C.; Freitas, S.R.; Rudorff, B.F.T.; Cortez, E.V.; Anselmo, E.; Carvalho, J.A. Pre-Harvest Sugarcane Burning: Determination of Emission Factors through Laboratory Measurements. Atmosphere 2012, 3, 164–180.
6. PM2.5/PM10 Particle Sensor Analog Front-End for Air Quality Monitoring Design (2016)
7. Fine Particulate Matter (PM-2.5) Simple Meter GuideBook (2018)
8. Yu, Zhou; Yu, Jun; Xiang, Chenchao; Fan, Jianping; Tao, Dacheng.: Beyond Bilinear: Generalized Multimodal Factorized High-Order Pooling for Visual Question Answering. IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, 2018, 29(12): 5947-5959.
9. Yu, Jun; Guo, Yukun; Tao, Dapeng; Wan, Jian.: Human pose recovery by supervised spectral embedding. NEUROCOMPUTING, 2015, 166: 301-308.
10. Yuyu Yin, Wenting Xu, Yueshen Xu, He Li, Lifeng Yu:Collaborative QoS Prediction for Mobile Service with Data Filtering and SlopeOne Model. Mobile Information Systems 2017: 7356213:1-7356213:14 (2017)
11. Yuyu Yin, Yueshen Xu, Wenting Xu, Min Gao, Lifeng Yu, Yujie Pei:Collaborative Service Selection via Ensemble Learning in Mixed Mobile Network Environments. Entropy 19(7): 358 (2017)

13. Y. Yin, W. Zhang, Y. Xu, H. Zhang, Z. Mai and L. Yu, QoS Prediction for Mobile Edge Service Recommendation with Auto-encoder, IEEE Access. doi: 10.1109/ACCESS.2019.2914737

13. Debnath Bhattacharyya, "Garlic Expert Advisory System using Parallel ABC Algorithm", International Journal of Disaster Recovery and Business Continuity, NADIA, ISSN: 2005-4289 (Print); 2207-6425 (Online), vol.8, November (2018), (pp. 11-20), http://dx.doi.org/10.14257/ijdrbc.2018.9.02.

14. N. Thirupathi Rao, "A Review on Industrial Applications of Machine Learning", International Journal of Disaster Recovery and Business Continuity, NADIA, ISSN: 2005-4289 (Print); 2207-6425 (Online), vol.8, November (2018), (pp. 1-10), http://dx.doi.org/10.14257/ijdrbc.2018.9.01.

15. G. M. Faruk Ahmed, "Systematic Literature Review: Disaster Recovery and Business Continuity Plan for Core Banking Solution (CBS) Project", International Journal of Disaster Recovery and Business Continuity, NADIA, ISSN: 2005-4289 (Print); 2207-6425 (Online), vol.8, November (2017), pp. 29-44, http://dx.doi.org/10.14257/ijdrbc.2017.8.03.
16. Ekbal Rashid, "Disease Detection on the Basis of Multiple Symptoms by Expert System", International Journal of Disaster Recovery and Business Continuity, NADIA, ISSN: 2005-4289 (Print); 2207-6425 (Online), vol.8, November (2017), pp. 1-10, http://dx.doi.org/10.14257/ijdrbc.2017.8.01.
17. Subhanil Guha and Anindita Dey, "Study on Morphological Changes in and around Nayachara Tail Using Remote Sensing Techniques", International Journal of Disaster Recovery and Business Continuity, NADIA, ISSN: 2005-4289 (Print); 2207-6425 (Online), vol.7, November (2016), pp. 13-26, http://dx.doi.org/10.14257/ijdrbc.2016.7.02.
18. Jun-hee Choi, Hyun-Sug Cho, "A Study on the Improvement of the Accuracy of Low-Cost Light Scattering Method Particulate Matter Sensors," Asia-pacific Society of Convergent Research Interchange, Jun., 2019.
19. J. B. Wedding and M. A. Weigand, "An Automatic Particle Sampler with Beta Gauging," Jour. of the Air Waste Manage. Assoc., vol. 43, no. 4, pp. 475-479, 1993.
20. source: ipac.caltech.edu
21. http://www.horiba.com/fileadmin/uploads/Scientific/Documents/PSA/BC002.pdf, 2007
22. C. Wu, “Implementing the Radix-4 Decimation in Frequency (DIF) Fast Fourier Transform (FFT) Algorithm Using a TMS320C80 D0 DSP,” Texas Instruments, Taiwan, Application Report: SPRA152, 1998.
23. H. Gao, Y. Duan, L. Shao, X. Sun. Transformation-based processing of typed resources for multimedia sources in the IoT environment. Wireless Networks, 2019. DOI: 10.1007/s11276-019-02200-6
24. Y. Zhu, W. Zhang, Y. Chen, H. Gao. A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment, EURASIP Journal on Wireless Communications and Networking, 2019, 2019(247). DOI:10.1186/s13638-019-1605-z
25. H. Gao, C. Liu, Y. Li, X. Yang. V2VR: Reliable Hybrid-Network-Oriented V2V Data Transmission and Routing Considering RSUs and Connectivity Probability. IEEE Transactions on Intelligent Transportation Systems, 2020, DOI: 10.1109/TITS.2020.2983835
26. H. Gao, W. Huang, Y. Duan. The Cloud-Edge Based Dynamic Reconfiguration to Service Workflow for Mobile Ecommerce Environments: A QoS Prediction Perspective. ACM Transactions on Internet Technology, 2020, DOI: 10.1145/3391198

**Jun-hee Choi** received the B.S. degree in Computer Engeenering from the Daejeon University, Daejeon, Korea, in 2016 and the M.S. degrees from the Chung-Ang University, Seoul, Korea, in 2019. Since March 2019, he has been with the Daejeon University, where he is Ph.D. student with the Department of Fire and Disaster Prevention College of Engineering. His research interests are Fire-fighting ICT, Computer Science and bigdata.

**Hyun-Sug Cho** received the B.S. degree in Mathmatics from the Daejeon University, Daejeon, Korea, in 1996 and the M.S. and Ph.D. degrees from the Daejeon University, Daejeon, Korea, in 2001 and 2008, respectively. Since March 2006, she has been with the Daejeon University, where she is Professor with the Department of Fire and Disaster Prevention College of Engineering. Her research interests are network security, fire-fighting ICT and bigdata.

*Received: September 05, 2019; Accepted: June 14, 2020*

Computer Science and Information Systems 17(3):891–906 https://doi.org/10.2298/CSIS190917026K

# A Load Balancing Scheme for Gaming Server applying Reinforcement Learning in IoT ⋆

Hye-Young Kim[1] and Jinsul Kim[2]

[1] School of Games, Major of Game Software,
Hongik University, Korea
hykim@hongik.ac.kr

[2] School of Electronics and Computer Engineering,
Chonnam National University, Gwangju, South Kore
jsworld@jnu.ac.kr

**Abstract.** A lot of data generated on the game server causes overtime in IoT environment. Recently, both researchers and developers have developed great interests in load balancing schemes in gaming servers. The existing literature have proposed algorithms that distribute loads in servers by mostly concentrating on load balancing and cooperative offloading in Internt of Things (IoT) environment. The dynamic load balancing algorithms have applied a technique of calculating the workload of the network and dynamically allocating the workload according to the network situation, taking into account the capacity of the servers. However, the various previous researches proposed are difficult to reflect the real world by imposing a lot of restrictions and assumptions on the IoT environment, and it is not enough to meet the wide range of service requirements for the IoT environment. Therefore, we proposed an agent that applies a deep reinforced learning method to distribute loads for gaming servers. The agent has accomplished this by measuring network loads and analyzing a large amount of user data. We specifically have chosen deep reinforcement learning because no labels would need to be obtained in advance and it enabled our agent to immediately make the right decisions to load balancing in IoT environment. We have showed several siginicicant functions of our proposed scheme and derived through mathematical analysis. Also, we have compared performances of our proposed scheme and a previus research, ProGreGA, widely used scheme through simulation.

**Keywords:** deep reinforcement learning, load balancing, gaming server, reward, achievable rate, loss rate, policy

## 1. Introduction

Recently, much study has been done applying load balancing to network [1]. Maintaining balanced workloads benefits the cloud service provider by increasing their resources utilization, eliminating the performance bottlenecks, and improving the quality of services to their customers. Load balancing schemes have been widely adopted by distributed servers and their effectiveness is of importance to the quality of services provided by such servers.

Load balancing has been studied using various approaches [2,3,4]. Centralized solutions are computationally extensive, require much information exchange overhead. To

⋆ Co-corresponding Author: Jinsul Kim

overcome these limitations, decentralized approaches have been proposed in citeRef4, and a distributed algorithms are used to solve it. However, most of these studies impose many restrictions and assumptions the networks that often do not apply in realistic networks [5]. Reinforcement Learning (RL) is a learning algorithm of decision the action to be performed in the learning system for maximizing reward to the action [6,7]. Due to the RL methods' advantages, it has been largely discussed in developing load balancing problems. In [8], RL was implemented for distributed load balancing in IoT network.

Therefore, we proposed a load balancing scheme applying RL method in IoT network. We address the key functions for the proposed scheme and simulate its efficiency using mathematical analysis.

The rest of the paper is organized as follows. Section 2 gives the previous reseaches related to open load balancing and reinforcement learning. In Section 3, we describes the detailed load balancing scheme of ours. In section 4, we describe the experimental results and show that the proposed scheme can effectively improve the performance for gaming server in IoT environment. In the final section, we constitute a summary of our proposal and suggest further study directions in IoT environment.

## 2. Related Works

The contributions of IoT depend on the increased value of information created by the number of interconnections among things and the subsequent transformation of processed information into knowledge for the benefit of society. Various researches use a clustering algorithm to utilize contextual information. In [9,10], the authors proposed a subvariance method based on neural regulation filtering by applying context information clustering and latent function learning fuzzy theory. After they have investigated similar neighbors of users and similar neighbors of services. When the clustering result is ready to learn the latent function of contextual information, join the potential node to the cluster [11].

Load balancing mechanisms are widely used in a distributed computing environment to balance the workloads on different servers, and the effectiveness of such mechanisms is critical to the overall performance and service quality. Load balancing can distribute workload across multiple entities to achieve optimal utilization, maximize throughput, minimize response time, and avoid overload. A lot of research has been done on how to design an effective load balancing such as in [12,13,14].

Deep learning has been applied to a many fields such as speech recognition, computer vision, natural language processing, social network filtering and bioinformatics. Deep learning is also applied when adopting multiple layers of nonlinear processing units for feature extraction and transformation [15]. The effect on deep learning can be guaranteed to be a universal approximation theorem, since this theory can be represented as a small subset of continuous functions in a feedforward network with a single hidden layer containing a finite number of neurons [16,17,18].

In [19],the authors have proposed a method that modeled a new neighbor feature learning method as a matrix by combining the advantages of a neighbor-based method, a model-based method, and a method based on deep learning. The proposed mothod was able to achieve high accuracy in neighborhood selection even with high data scarcity, and was able to learn deep features. The learning systems they limit use a learning convolutional neural network to learn deep learning from the selected neighbor's cell record, and also learn the relationship with the features of the target user or target service.

The RL is a different from supervised learning in that it doesn't need input/output pairs. This focuses on performance, which involves finding the balance between explorations. This learning system creates $a_1, a_2...a_n$ actions to interact with the environment. These actions affect the environmental condition, and as a result, the RL system receives scalar rewards $r_1, r_2...r_n$. The goal of this learning system is to learn how to act in a way that maximizes future rewards through learning. The RL approaches store the results of interaction with the past environment and find the optimal policy for repetitive learning [20].

RL could be applied as a method for making optimal decisions. The agent for this has taken into account the environment. At every step, the agent has taken action and receives observations and rewards. RL algorithms has tried to take full account of a given, previously unknown environment. RL made choices to maximize rewards in each stage of learning, and learned the policy to find the maximum reward value by repeating the steps. They have been applied to many different fields. The policy optimization method used the policy of each step to map the agent's state to the next action and learns by reflecting the result value in the next step. These methods showed RL as a numerical optimization method. We could optimize the expected rewards for an efficient learning system in relation to the parameters of the policy.

The challenges herein are to consider a priori how many interactions are important to learn a specific task and what exact features should be extracted. Deep neural networks are the quintessential technique for automatic feature extraction in reinforcement learning [21,22]. Also, various previous researches in load balancing had not effectively taken into account the rapidly increased event and uncertainly status for gaming server in IoT network.

Therefore, we proposed a load balancing scheme applying reinforcement learning in order to efficient load balancing in IoT environment.

## 3. Proposed Scheme

### 3.1. System Configuration

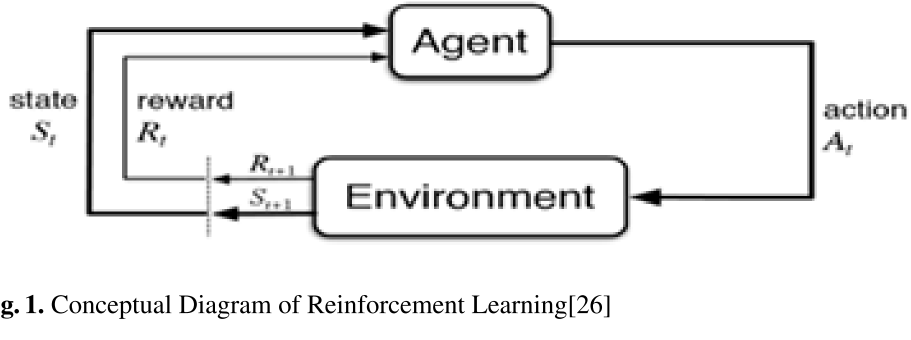


**Fig. 1.** Conceptual Diagram of Reinforcement Learning[26]

RL is one of a machine learning used to automate goal learning and decision making. Fig. 1 has been showned to the concept of RL method applied to the our scheme in this paper. When the agent in the proposed system received input, in the current state $s$, the agent performs the corresponding action $a$. As the result, the reward value $r$ has provided. Based on the reward value of $r$, the learning system transmitted to the new state $s^{'}$ and the agent processed again as $a^{'}$. Depending on its current rewards and status, the reinforcement system has chosen the next action based on a policy that increases the likelihood of agent positive rewards. The goal of RL agents is to maximize the total rewards received from the proposed system to find the optimized policy.

### 3.2. Network Load Learning Algorithm

The RL algorithms are generally applied to obtain optimal results by adjusting the motions in observed state of discontinuous and low-dimensional motions [23]. However, along with the development of computing capacity and deep learning, a new algorithm called Deep Reinforcement Learning (DRL) has appeared. In order to model complex nonlinear relationships such as IoT networks, we have applied RL to map the structure of the network load. In IoT networks, load balancing is only achieved in a small local area, so the network area is divided into several smaller areas based on the zone. Our proposed load balancing scheme is by system configuration applying DRL algorithm for distributed static load balancing of gaming server. For the mathematical modeling of our proposed scheme, we have used the following algorithm. The Variables used to model our proposed scheme have shown in Table 1.

As each base station generally have been served a large number of user nodes, the important metric for network performance has the rate of service speed, not Signal to Interference Plus Noise Ratio (SINR) [24,25]. The rate of service speed being experienced

**Table 1.** parameters of modeling

| Parameters | Descriptions |
|---|---|
| $B$ | base station |
| $N$ | nodes of user |
| $P_j$ | power of base station, j |
| $\sigma$ | noise power |
| $t_0$ | start time |
| $r_j$ | reward of j |
| $Z$ | zone area |
| $w$ | weight of the node or server or cell) |
| $S_j$ | state of j node in IoT |
| $\pi$ | map states |
| $\beta$ | control value for learning |
| $\lambda$ | learning rate |
| $A$ | action space |

by the user node depends on the network load. We have defined the service speed rate as $R_{ij}$ and the achievable rate as $c_{jj}$ for independent of channel qualities. Let $B$ be the set of base stations with more than one node in network and that share resources and $N$ be the node of users. Let $P_j$ be a power of base station $j$ and $\sigma^2$ be a noise power level and $G_{ij}$ be a channel gain between $i$ node and $j$ base station. Therefore, $\sum_{k\in B,k=j} P_k G_{ik}$ has shown the interferences in the network. Let $t_0$ be a start time and $t$ as a present time and $\tau(t_0 \leq \tau \leq t)$ as a variable respresenting time, respectively. Let $F_{ij}(t)$ be a fraction time of resource that is the base station $j$ servers node $i$ and $H_{ij}$ be a long term service rate. Let $x_{ij}\tau$ be a scheduling indicator.

$$c_{ij} = \log_2(1 + SINR_{ij} = \log_2(1 + \frac{P_j G_{ij}}{\sum_{k\in B,k=j} P_k G_{ik} + \sigma^2}) \tag{1}$$

$$H_{ij}(t) = F_{ij}(t)\int_{t_0}^{t} x_{ij}(\tau)C_{ij}(\tau)d(\tau) \tag{2}$$

For efficient load balancing, the larger the value of $\sum_{j\in B}\sum_{i\in N} H_{ij}(t)$ and the smaller the variance of user's service rate $H_{ij}(t)$ should be. We have derived the results by constructing an estimator of our system using importance sampling in a large and continuous state. Our scheme have allowed to make decisions for each process according to a learned policy without wasting time for complex calculations. It is also possible to determine the optimal action based on the reward value of each phase without accurate information on the reward value or probability value of all environments. The agent of ours has learned to output the desired result value by input using present input and output data sets. Our network load learning algorithm has calculated and stored the current network load and learned the result value effectively.

**Algorithm 1** Load Balancing Agent applying Reinforcement Learning

```
for $i$ = 1, 2,... $S$ do do
    for $j$ = 1, 2, ... $N$ do do
        choose the best decision value, $argmax_j D_{ij}(t)$
        calculate its current reward, $r_j(t)$ based on 1
        calculate long term average reward of set $S_i^j$, $j$ is different set of actions
    end for
    set allocation of servers for optimize load balancing
end for
```

Let $r_j$ be denote the reward of a phase $j$, $S$ be indicate the status, and $A$ be denote the behavior. Therefore, the $S_j$ is the phase in $j$ where $j$ is the station. Given each phase, $s$ , it mapes directly to the determined action $a$. Every $a \in A(s)$ has a probability distribution or could be deterministic $\pi(s)$. The policy for load balancing, that is the action determined by the state $s$ or stochastic $pi(a \mid s)$. In order to achieve efficient load balancing, RL has applied in our system, and a $pi$ policy has developed to select possible behaviors in each phase and map behavior to state that improve $pi$ to be optimal. Load balancing policies could be either stochastic $\pi(a \mid s)$, which given a state $s$, each action $a \in A(s)$ is a probability distribution, or deterministic $\pi(s)$, that has maped a state, $s$, to a determined action, $a$. We have calculated the reward of base station $j$ as $r_j$.

$$r_j = \frac{1}{\sum_i^{S_j} \frac{1}{s_j} \cdot (R_{ij} - \frac{\sum_{k\equiv1}^{|B|} \sum_{i\equiv1}^{|V|} R_{ik}}{|V|})^2} \tag{3}$$

Let $AL_{ij}(t)$ be the allocation priority of node $i$ at the base station $j$ and $C_{ij}$ is broadcast at achievable rate.

$$AL_{ij}(t) = \frac{C_{ij}(t)}{R_{ij}(t-1)} \tag{4}$$

We have $pi$ to represent the probabilistic policy $\pi$: $S$ X $A \longleftarrow$ [0, 1], and the expected discount compensation for $\eta$ ($\pi$) to indicate. We have considered the policy $\pi_\theta(a|s)$ with parameter vector $\theta$, and we have used function of $\theta$ rather than $overloaded.\pi$, $L_\theta(\tilde{\theta})$: = $L_\pi(\pi_{\tilde{\theta}})$ and $D_{KL}$ ($\theta \parallel \tilde{\theta}$): = $D_{KL}$ ($\pi_\theta \parallel \pi_{\tilde{\theta}}$). We used $\theta_{old}$ as a parameter to improve the previous policy in our scheme. We have sampled $s_0 \sim \rho_0$ and simulated the $\pi_{\theta_i}$ policy. Then, following these trajectories of $s1, s2, ..., s_m$, have selected a subset of $N$ states. The agent $\hat{Q}_{\theta_i}$ ($s_m$, $a_{m,k}$) for each of the action have sampled from each the status $s_m$ and $a_{m,k}$ and processed actions in that phase.

$$L_m\theta = \sum_{k=1}^{k} \pi_\theta(a_k \mid s_m)\hat{Q}(s_m, a_k) \tag{5}$$

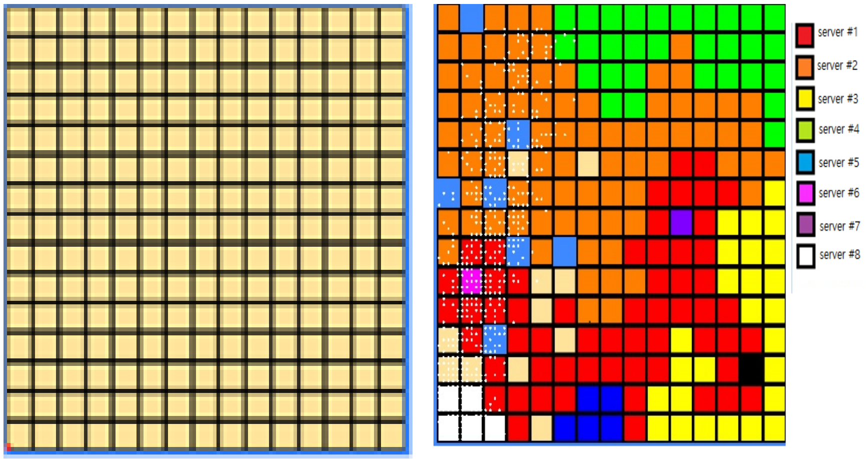
We have generated for every possible action in the state of that phase given in each of the phase. In our scheme, the agent have processed the $a_{n,k}$action as $K$ behavior in each phase state, $s_n$, as represented $a_{n,1}, a_{n,2}, ..., a_{n,k}$. The results obtained in each pahse have represented as $L_{\theta_{old}}$. We have estimated $L_{\theta_{old}}$ from the expectation and gradient of $s_n \sim \rho(\pi)$ for $L_{\theta_{old}}$.

$$L_m(\theta) = \frac{\sum_{k=1}^{k} \frac{\pi_\theta(a_{m,k}|s_m)}{\pi_{\theta_{old}(a_{m,k}|s_m)}} \hat{Q}(s_m, a_{m,k})}{\sum_{k=1}^{k} \frac{\pi_\theta(a_{m,k}|s_m)}{\pi_{\theta_{old}(a_{m,k}|s_m)}}} \tag{6}$$

## 4. Performance Analysis

### 4.1. Simulation

In this study, we have used ML-agents library from Unity3D to simulate the load balancing agent for gaming server applying our proposed scheme at Section 3. The learning is processed by the repeated cycle of sending variables by Tensor-flow which are collected from learning environments created by Unity3D and sending back results which are learned from Proximal Policy Optimization (PPO) algorithms, one of the RL. We have set the hidden layer of neural network to 3 and the node of a hidden layer to 256 as refered in [27,28]. Also, we have set the size of batch to 512 and the $\beta$ value to control the entropy to $\log_e^{-3}$. In addition, we have set the learning step to 5 million steps for our simulation environment.



(a) Simulation environment

(b) Initial State

**Fig. 2.** Initial Simulation environment

It consists of a two-dimensional map of the game world, each of that contained a Finite State Machine(FSM) in the place of 750 gaming users. The weight of each bot is reset to 1. We have assumed that there are 8 servers and used policy that load balancing agent learned, to disperse the load. Fig.2(a) is shown our environment of simulation.

For analysis of load balancing result, we have differentiated each server by color as shown in Fig. 2(b). Each rectangle represents a cell, and a group of areas is defined as an area, and a group of the areas is defined as a world. Each server is given an area.

The game world is made up of 15*15 grid world and has 225 cells which are allocated with 8 servers. User load is occurred by activating 750 users, all which are processed by FSM. Also, server capacity is defined as i*20000 and i is a value between 1 to 8. Therefore, we have assumed the capacity of a server 1, 2, and 3 as 20000, 40000, and 160000. Fig. 2(b) is shown the initial state.

## 4.2. Performance Analysis

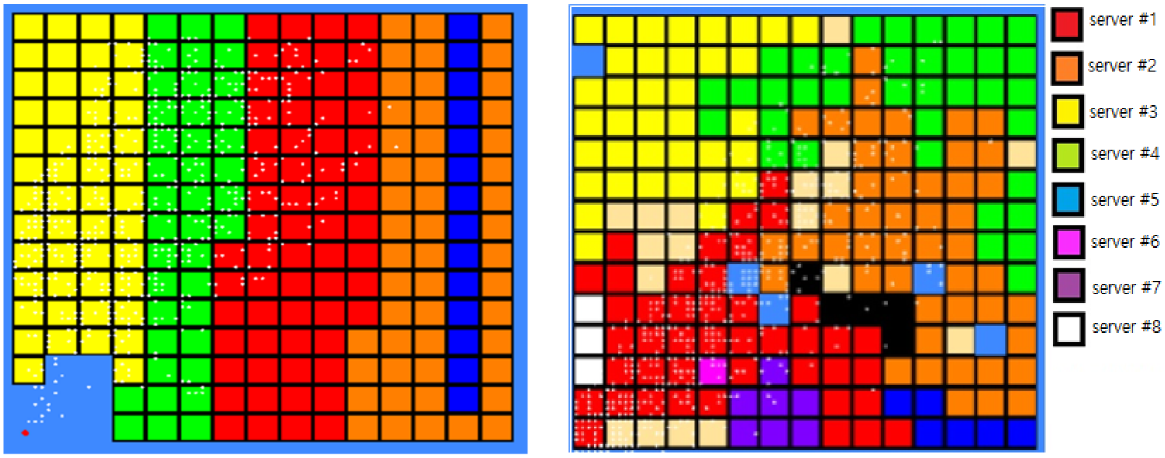
In the experiment, a standard that we have set are as follow: 1) The weight of 750 users has been set as 1
2) After defining the weight, we set a number of users between 500 to 1000
3) After defining the number of users, we multiply the weight by a value between 0.5 to 2

It was showed in [28] that the ProGreGA algorithm has numerous advantages compared to other load-balancing algorithms such as BFBCT, Kernighan-Lin, and Ahmed such as fewer walk migrations, minimized overhead, and the maintenance of the maximum possible number of cells when rebalancing, resulting in the ProGReGA algorithm having the most efficient all the simulated algorithms.

**Algorithm 2** ProGReGA Load Balancing Algorithm

```
initiate $Weight_{Division}, Capacity$
for each zone $z$ in zone list $Z$ do do
    $Weight_Division = Weight_{Division} + W_z(Z)$
    $Capacity = Capacity + Y(S_Z)$
end for
sort zone list in decreasing $Y(S_Z)$
for each zone $z$ in zone list $Z$ do do
    $Weight = Weight_{Division} \times \frac{Capacity}{Weight_{Division}}$
    while $W_z(Z) \leq Weight$ do do
        if any cell from $Z$
        $Z = Z \cup$ with heihest cell in the zone
        else
        $Z \cup$ the cell
    end while
end for
```

Therefore we will compare our proposed scheme only with the ProGReGA algorithm. The Alorithm 2 is shown the ProGReGA methods as in [28], and we have experiment based on 2 to simulate the ProGReGA.



(a) Proposed Scheme

(b) Previous(ProGReGA) Scheme

**Fig. 3.** Result of load balancing in simulation environment

In the propose scheme, the average fragmented cells occurred is 2 or 3 times more than that in the ProGReGA. A result of 100 experiments is sorted in ascending order of fragmented cells and divided it in three to indicate which are the worst, average, and best results.

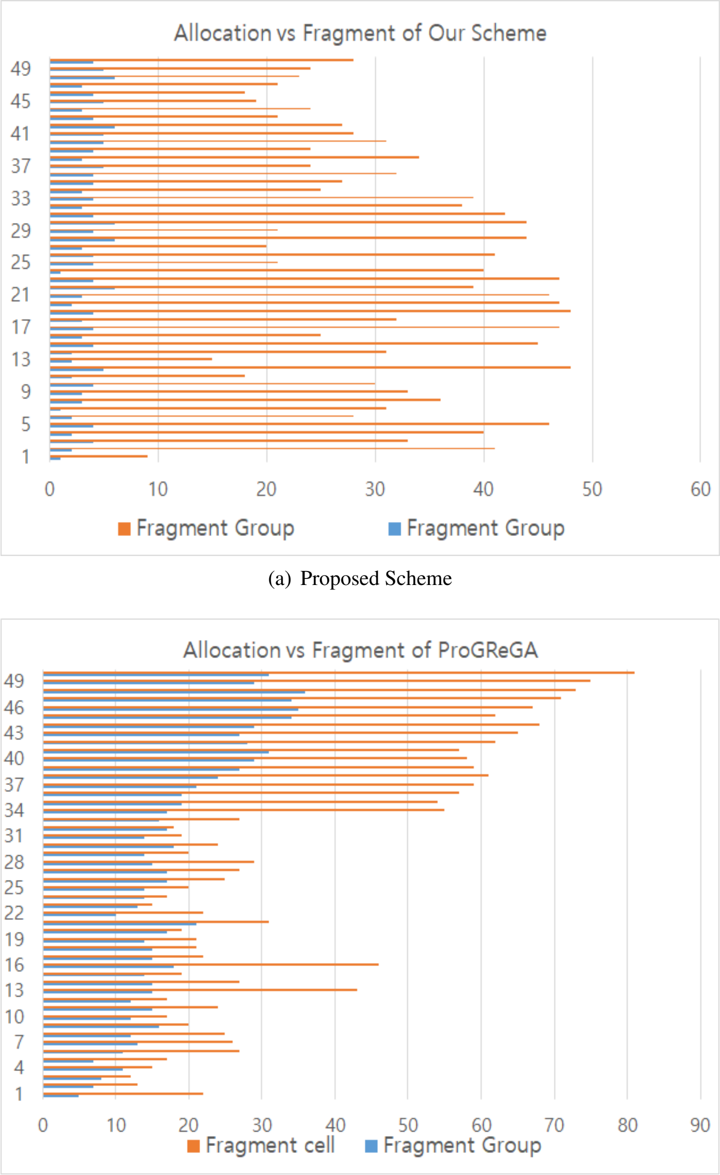
We have put a criterion that user one load interacts with all users, and to confirm the result of balancing, we have dispersed the load in a way that is not affected by the previous load balancing result. To check the process of dispersion, we have set a color for each server and alter the color of the dispersed cell to the corresponding server color. We have selected a scenario that where each user randomly selects bearing and move to that direction.

When an agent chooses a cell that hasn't been selected before, it gets +1 as a reward, otherwise, it gets -0.5 as a reward. In addition, if it selects a cell which is not near the cell it is currently located, it gets -0.1 as a reward. Thus, when one episode ends, we add up the reward it got and additionally add it with the value of the standard deviation of dispersed server usage, multiply with 10. It means that as usage standard deviation decrease, distribution of usage is constant, i.e., usage of all server is distributed equally

The Fig.3(a) and Fig.3(b) are shown the result of the load balancing of our proposed scheme and previous scheme, respectively.

The experimental result of a model that has been learned through our scheme shows 75% increase in its performance compared with the previous research, ProGreGA. The occurrence aspect of the conditional fragmented cell are shown no much difference from ours. However, it could be realized that the occurrence rate for the fragmented cell is too much and change in the occurrence rate changed drastically.

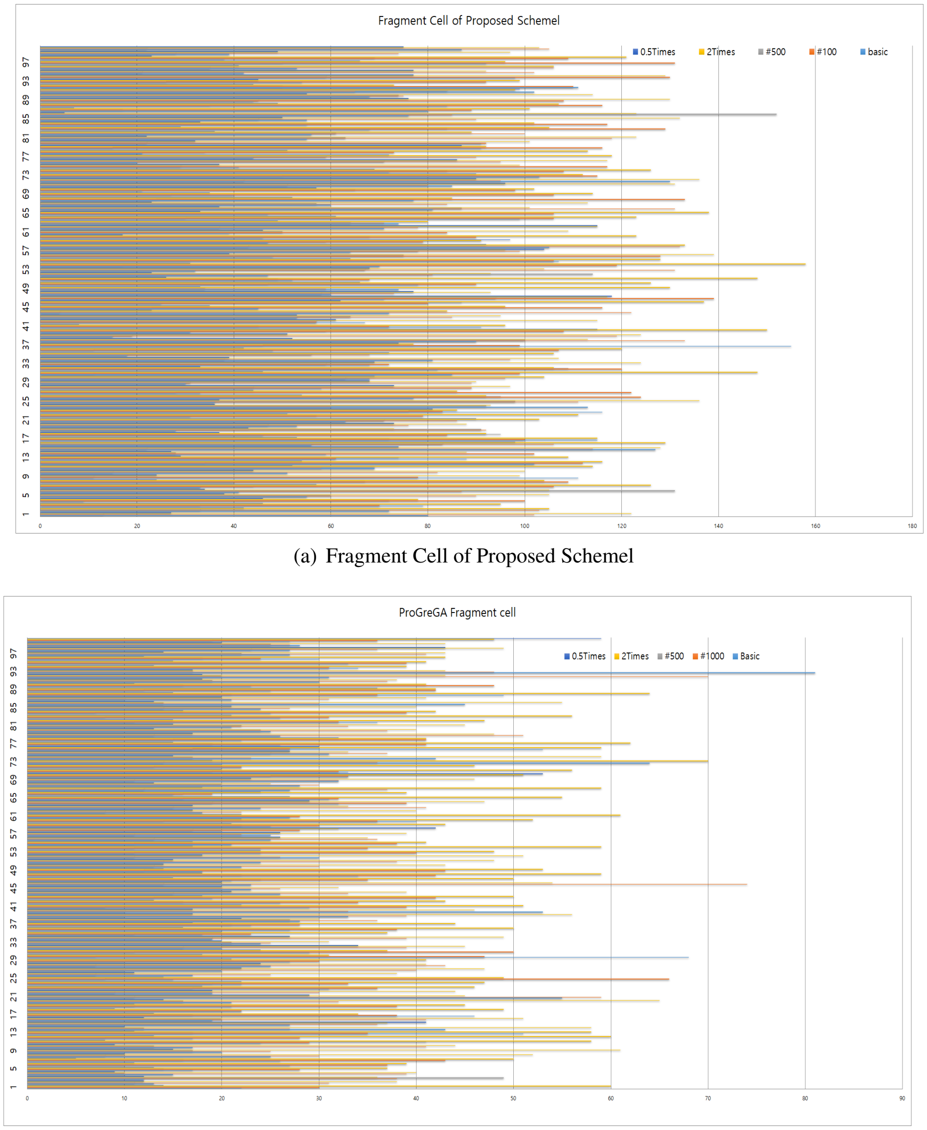
As shown in Fig. 4, our proposed scheme have showed better performance than previous research, ProGreGA. The conditional fragment cell of model that has been learned through the same condition is shown as a graph in Fig. 5.



(a) Proposed Scheme

(b) Previous(ProGReGA) Scheme

**Fig. 4.** Result of allocation and fragement cells of server

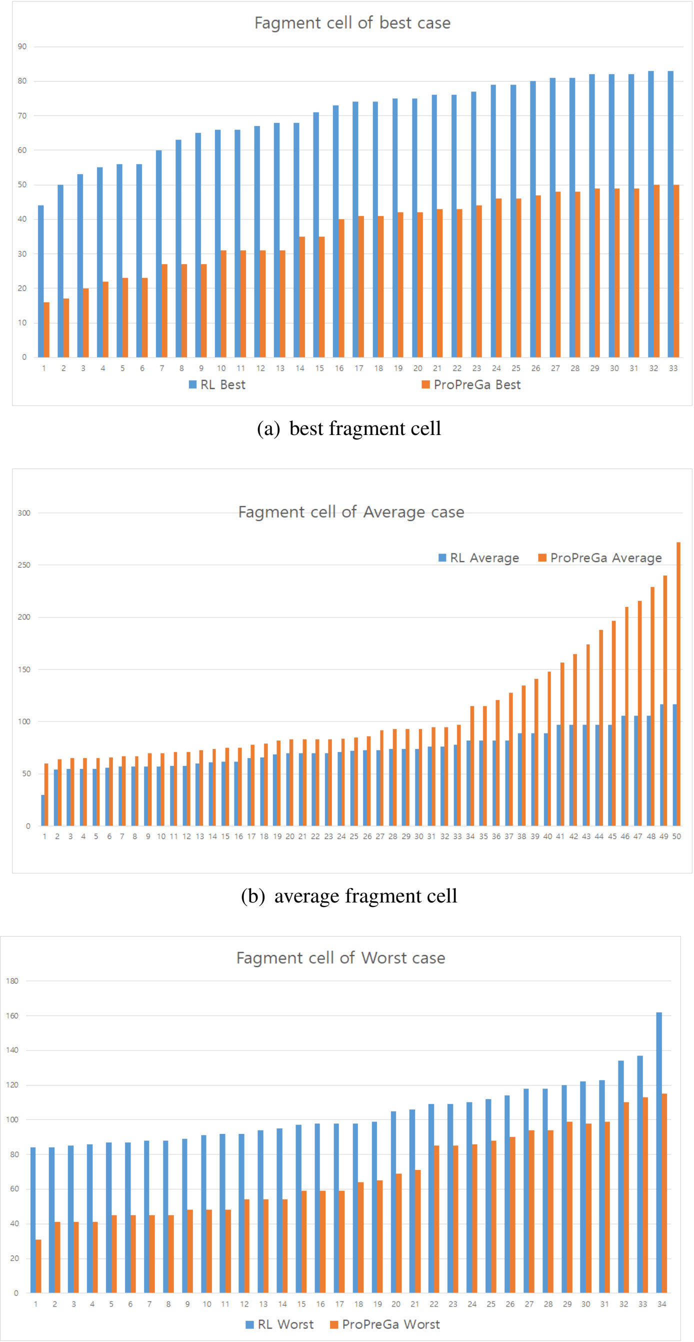


(a) Fragment Cell of Proposed Schemel

(b) ProGreGA Fragment cell

**Fig. 5.** Fragement cells after Load balancing

Also, our proposed scheme have shown no significant change in the occurrence rate in case of average and worst at the simulation as shown Fig.6(b) and Fig. 6(c). As shown the Fig.6(a), our proposed scheme have shown no rapid change in the occurrence rate just like in average result.



(a) best fragment cell

(b) average fragment cell

(c) worst fragment cell

**Fig. 6.** Result of fragment cell after load balancing each of gaming server

We have simulated 100 experiments on load balancing under various conditions, and the results have been shown in Fig.7. The more fragmented cells that are not allocated to servers for load balancing, the lower the performance. The reason why the number of occurrences of the fragmented group is important because servers are assigned based on the group unit.

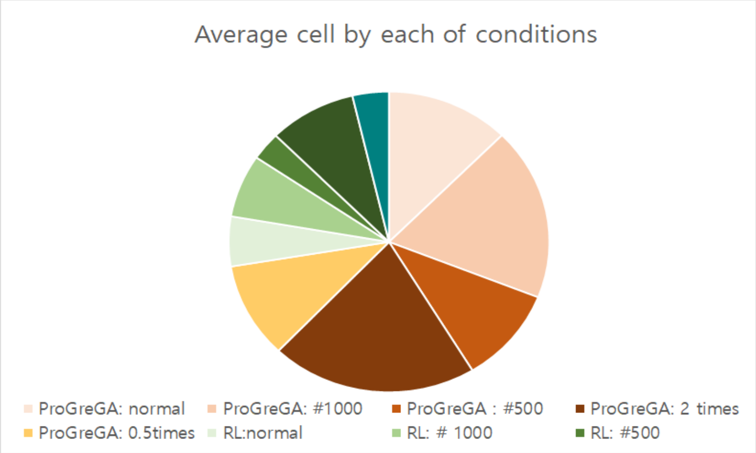


**Fig. 7.** Result of 100 executin fragment cell by conditions

## 5. Conclusion and future works

In this study, we have proposed an agent that applies a deep reinforced learning method to distribute static loads for gaming servers. We have addressed several key functions of our proposed scheme and derived the efficiency of ours through mathematical analysis. The agent has been accomplished this by measuring network loads and analyzed the large amount of user data and network loads, all with the aforementioned DRL.

We have used ML-agents library from Unity3D to simulate the load balancing agent for gaming server applying our proposed scheme. The learning was processed by the repeated cycle of sending variables by Tensor-flow collected from learning environments created by Unity3D and sending back results that are learned from Proximal Policy Optimization (PPO) algorithms, one of the reinforced learning. We have simulated 100 experiments on load balancing under various conditions, where, the more fragmented cells not allocated to servers for load balancing, the lower the performance. The number of occurrences of the fragmented group is important because servers are assigned based on the group unit. We compared the performance of the ProGreGA algorithm which was shown to be the most efficient among the previous research as in [28] with our proposed scheme by running mathematical modeling and simulations. The simulation result of a model learned through our scheme has been shown 75% increase in its performance compared with the ProGreGA. The occurrence aspect of the conditional fragmented cell has been shown no much difference from ours. However, the occurrence rate for the fragmented cell was too much and the occurrence rate changed drastically. Our proposed scheme have shown the efficiency of load balancing and it is required further works reflect real world in network.

In the future, we intend to evaluate performances by collecting data from applying the proposed scheme in the real world, such as in game servers and blockchain platforms. In addition, we would analyze the collected data and analyze performance through various deep learning algorithms.

**Acknowledgments.** This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1A2C1008533) and this research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT and Future Planning(No. 2016RIA2B4012386) and this work was supported by 2020 Hongik University Research Fund.

## References

1. Ahmed D., Shirmohammadi S.: A microcell oriented load balancing model for collaborative virtual environments. In: Proceeding of the IEEE International Conference on Virtual Environments, Human Computer Interfaces and Measurement Systems, VECIMS, pp.86-91 (2008)
2. Andrews J., S. Singh, Q., Lin X.,Dhillon H.: An overview of load balancing in HetNets: Old myths and open problems. In: IEEE Wireless Com-mun., 21(2), 18–25, Apr. (2014)
3. O. K. Tonguz, Yanmaz E.: The mathematical theory of dynamic load balancing in cellular networks. In: IEEE Trans. Mobile Comput. 7(12), 1504–1518, Dec (2008)
4. H. Kim, Veciana G., Yang X., Venkatachalam M.: Distributed alpha-optimal user association and cell load balancing in wireless net- works. In: IEEE/ACM Trans. Netw., 20(1), 177–190, Feb. (2012)
5. Zhang Z., Ma L.,, Leung K., Tassiulas L., Tucker J.: Q-placement: Reinforce-ment-Learning-Based Service Placement in Software-Defined Networks. In: Proceeding IEEE International Conference on Distributed Computing Syst. (ICDCS), 1527-1532, July (2018)
6. Sutton R., Barto A.: Reinforcement Learning: An Introduction. In: Cambridge, MA, USA: MIT Press, (1998)
7. Honghao G., Kuang L., Yin Y., Guo B., Dou K,: Mining Consuming Behaviors with Temporal Evolution for Personalized Recommendation in Mobile Marketing Apps. In: ACM/Springer Mobile Networks and Applications (MONET), 2020, DOI: 10.1007/s11036-020-01535-1
8. Parent J., Verbeeck K., Lemeire J.: Adaptive Load Balancing of Parallel Applica-tions with Reinforcement Learning on Heterogeneous Networks. In: Interational Symposeum Distributed Computing and Application for Bussiness Enginering and Science, 16-20 Dec. (2002)
9. Honghao G.,Yueshen X., Yuyu Y., Weipeng Z., Rui L., Xinheng W.: Context-aware QoS Prediction with Neural Collaborative Filtering for Internet-of-Things Ser-vices. In: IEEE Internet of Things Journal, (2019), https://doi.org/10.1109/JIOT.2019.2956827.
10. Xiaoxian Y., Sijing Z., Min C.: An Approach to Alleviate the Sparsity Problem of Hybrid Collaborative Filtering Based Recommendations: The Product-Attribute Perspective from User Reviews. In: MOBILE NETWORKS and APPLICATIONS, 25(2), 376-390 (2020)
11. Honghao G., Wanqiu H., Yucong D.: The Cloud-Edge Based Dynamic Reconfiguration to Service Workflow for Mobile Ecommerce Environments: A QoS Prediction Perspective. In: ACM Transactions on Internet Technology (2020), DOI: 10.1145/3391198

12. Hye-Young K. : A Load Balancing Scheme with LoadBot in IoT Networks. In: Journal of Supercomputing, 74(1), 1215-1226, July (2017), DOI 10.1007/s11227-017-2087-6.
13. Deng S., Xiang Z., Zhao P., Taheri J., Hinghao G., Yin J., Zomaya Y,: Dynamical resource allocation in edge for trustable iot systems: a reinforcement learning method. In: IEEE Transactions on Industrial Informatics, (2020), DOI: 10.1109/TII.2020.2974875
14. Honghao G., Liu C., Youhuizi L., Xiaoxian Y.: V2VR: Reliable Hybrid-Network-Oriented V2V Data Transmission and Routing Considering RSUs and Connectivity Probability. In: IEEE Transactions on Intelligent Transportation Systems, (2020), DOI: 10.1109/TITS.2020.2983835
15. Jinglin L., Guiyang L., Nan C., Quan Y., Zhiheng W., Shang G.,Zhihan L.: An End-to-End load Balancer Based on Deep Learning for Vehicular Network Traffic Control. In: IEEE Internet of Things Journal, 6(1), pp.953-966, February (2019)
16. Cybenko G.: Approximation by superposition of a sigmoidal function. In: Math. Control Signals Syst., 2(4), 303–314 (1989)
17. Jun Y., Chaoqun H., Yong R., Dacheng T.: Multitask Autoencoder Model for Recovering Human Poses. In: IEEE Transactions on Industrial Electronics, 65(6), 5060-5068 (2018)
18. Jun Y., Zhenzhong K., Baopeng Z., Wei Z., Dan L., Jianping F.: Lever-aging Content Sensitiveness and User Trustworthiness to Recommend Fine-Grained Privacy Settings for Social Image Sharing. In: IEEE Transactions on Information Forensics and Security, 13(5), 1317 – 1332 (2018)
19. Yuyu Y., Lu C., Yueshen X., Jian W., He Z., Zhida M.: QoS Prediction for Service Recommendation with Deep Feature Learning in Edge Computing Environ-ment. In: Mobile Networks and Applications, (2019), https://doi.org/10.1007/s11036-019-01241-7.
20. Revar A., Andhariya M., Sutariya D.: Load Balancing in Grid Environment using Machine Learning-Innovative Approach. In: International Journal of Computer Applica-tions, 8(10), 31-34, October (2010)
21. Sutton R., Barto A.G.: Reinforcement Learning: An Introduction. In: Cambridge, MA: MIT Press (1998)
22. Bertsekas D.P.,Tsitsiklis J.N.: Neuro-Dynamic Programming. In: Nashua, NH, Athena Scientific (1996)
23. Goodfellow I., Bengio Y., Courville A.: Deep Learning [Online]. In: MIT Press (2016), http://www.deeplearningbook.org
24. Cano G.L., Ferreira M., Simoes A.S., Luna C.: Intelligent Control of a Quadrotor with Proximal policy Optimization Reinforcement Learning. In: Lat-in America Robotic Symposium, pp.503-508 (2018)
25. Ye Q., Rong B., Chen Y., Al-Shalash M., Caramanis C., Andrews J.: User as-sociation for load balancing in heterogeneous cellular networks. In: IEEE Transactions Wireless Communications, 12(6), 2706–2716, Jun (2013)
26. Richard S.: Reinforcement Learning:An Introduction. In: Cambridge, P.48 (2018)
27. Andrews J., Singh s., Ye Q., Lin A., Dhillon H.: An overview of load balancing in Het-Nets: Old myths and open problems. In: IEEE Wireless Communications, 21(2), 18–25, Apr. (2014)
28. Carlos E., Cláudio G.: A Load Balancing Scheme for massively multiplayer online games. In: Multimedia Tools and Applications, 45(1), 263-289, October (2009)

**Hye-Young Kim** received the Ph.D. degree in Computer Science and Engineering from the Korea University of South Korea in February 2005. During her Ph.D. study, she had focusing on location management scheme and traffic modeling, such as mobile IPv6, cellular network and network mobility. Currently, she works at Hongik University of South

Korea as a Full Professor since March 2007. She had developed a network protocol for 9 years while she was working at Hyundai Electronics as a senior researcher. Her research interests include traffic modeling and load balancing scheme applying deep learning in IoT and Block-Chain.

**Jin-Sul Kim** received the B.S. degree in computer science from The University of Utah, Salt Lake City, UT, USA, in 2001, and the M.S. and Ph.D. degrees in digital media engineering from the Department of Information and Communications, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2005 and 2008, respectively. He worked as a Researcher with the IPTV Infrastructure Technology Research Laboratory, Broadcasting/ Telecommunications Convergence Research Division, Electronics and Telecommunications Research Institute (ETRI), Daejeon, from 2005 to 2008. He worked as a Professor with the Korea Nazarene University, Chonan, South Korea, from 2009 to 2011. He is currently a Professor with Chonnam National University, Gwangju, South Korea. His research interests include cloud computing, MEC, smart factory/city based Intelligent application development, and intelligent networking solutions.

*Received: September 19, 2019; Accepted: May 8, 2020.*

Computer Science and Information Systems 17(3):907–926 https://doi.org/10.2298/CSIS190908027Y

# Graph Embedding Code Prediction Model Integrating Semantic Features

Kang Yang[1], Huiqun Yu[1,2], Guisheng Fan[1,3], and Xingguang Yang[1]

[1] Department of Computer Science and Engineering, ECUST
Shanghai, China
[2] Shanghai Key Laboratory of Computer Software Evaluating and Testing, China
Shanghai, China
yhq@ecust.edu.cn, Corresponding author
[3] Shanghai Engineering Research Center of Smart Energy, Shanghai, China
Shanghai, China
gsfan@ecust.edu.cn, Corresponding author

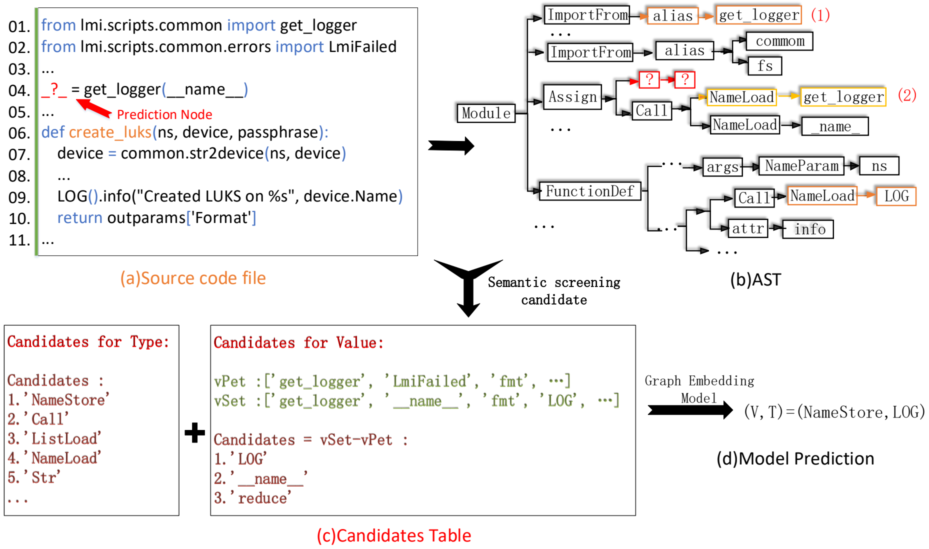
**Abstract.** With the advent of Big Code, code prediction has received widespread attention. However, the state-of-the-art code prediction techniques are inadequate in terms of accuracy, interpretability and efficiency. Therefore, in this paper, we propose a graph embedding model that integrates code semantic features. The model extracts the structural paths between the nodes in source code file's Abstract Syntax Tree(AST). Then, we convert paths into training graph and extracted interdependent semantic structural features from the context of AST. Semantic structure features can filter predicted candidate values and effectively solve the problem of Out-of-Word(OoV). The graph embedding model converts the structural features of nodes into vectors which facilitates quantitative calculations. Finally, the vector similarity of the nodes is used to complete the prediction tasks of TYPE and VALUE. Experimental results show that compared with the existing state-of-the-art method, our method has higher prediction accuracy and less time consumption.

**Keywords:** Big Code, Graph Embedding, Code Prediction.

## 1. Introduction

With the rapid increase of the code's volume, the use of existing code data for prediction has attracted more and more attention. It makes use of the existing code in the context to suggest the next possible code token, such as method calls or object fields. However, the source code files are written by different programmers and have no fixed structure. For example, Python has a more casual programming style, and feature extraction process is more difficult than Java. These reasons lead to the low accuracy of the prediction model, and we may even get the opposite result. Thus, mining the information between nodes can effectively extract the features in the source code file. Finally, the model can predict the missing nodes through these features. Traditional code prediction methods are based on code contexts and grammatical rules. Early research focused on probability models in this field. The probability model integrates the node information around the predicted node and calculates the probability value of each candidate value in the model. For example, the N-gram[5] model uses the $n-1$ tokens to predict the probability of the $nth$ token. The N-gram model has strong flexibility and high scalability. Yet, N-gram model only extracts code information in a small range, and cannot deal with the long-distance dependent features of the code. So, the prediction performance of the N-gram model has limitations. Deep learning can effectively extract the features of long-distance node information. These methods have long been used in natural language processing tasks, such as LSTM. Raychev[24] believes that programming languages are also natural languages, such as recurrent neural network (RNN) models can also solve code problems. They uses the RNN model to obtain the order distribution and semantics of the code from the AST of the source code file. Neural language model combined with Attention Mechanism[4] can complete the code prediction task. However, the time consumption of neural network models is very long, and the interpretability of the model is weak. The neural network model uses fixed candidate values, so it is difficult to solve the OoV problem.

Generally, the last component of the neural network model is the softmax classifier. Each output dimension corresponds to a unique word in a predefined vocabulary. Due to the large amount of calculation of the model, the usual approach is to use only the $K$ most frequent words in the corpus to build a vocabulary. The words that are not in the candidate value are defined as Out-of-Vocabulary(OoV) words. In reality, every programmer will artificially define rare node names. The occurrence of these values is very small in the source file, and the probability of appearing in the candidate value table is very low. Therefore, OoV problems are common in code prediction, which also reduces the prediction accuracy of the model. Abstract Syntax Tree(AST) is an abstract representation of the grammatical structure of source code. It shows the grammatical structure of the programming language in the form of a tree, each node on the AST represents a structure in the source code. Each code source file has only one corresponding abstract syntax tree. We extract the rich grammatical structure features in AST and bring them to the downstream prediction model.



```
from lmi.scripts.common import get_logger
from lmi.scripts.common.errors import LmiFailed
...
_?_ = get_logger(__name__)
...
def create_luks(ns, device, passphrase):
    device = common.str2device(ns, device)
    ...
    LOG().info("Created LUKS on %s", device.Name)
    return outparams['Format']
...
```

Candidates for Type:

Candidates :
1.'NameStore'
2.'Call'
3.'ListLoad'
4.'NameLoad'
5.'Str'
...

Candidates for Value:

vPet :['get_logger', 'LmiFailed', 'fmt', …]
vSet :['get_logger', '__name__', 'fmt', 'LOG', …]

Candidates = vSet-vPet :
1.'LOG'
2.'__name__'
3.'reduce'

**Fig. 1.** Examples of Python programs and their corresponding AST

In this paper, we propose a graph embedding prediction model with AST's semantic structure features. We extract semantic structural features from the code context and filter out irrelevant candidate values. This process can narrow the range of candidate values, thereby shortening the prediction time. And the candidate values change according to the context of the prediction files. The dynamic change of candidate values can effectively solve the OoV problem. Besides, we extract the AST's node paths of the source file and convert them into training graphs. Then, we use the Node2vec model to extract the structural features of the graph and embed the structure paths into vector with fixed dimensions. The embedding model reduces the dimensionality of discrete node paths to continuous dimensional vectors, which is useful for quantitative calculations in downstream tasks. The Node2vec model biasly converts the training graph into node sequences. Each node sequence merges different structural information between nodes through parameters $(p, q)$. The greater the overlap degree of node sequences, the higher the similarity of nodes. Therefore, the probability of our predicted value is positively correlated with the similarity of the node's structure. Our model can keep the structural information of the source code, so the model interpretability is higher. Besides, the model consumes less time than the neural network models and has higher prediction accuracy.

As shown in Fig. 1, the source code file in Fig. 1(a) is got from GitHub. The **Prediction Node** in the Fig. 1(a) is the node we want to predict. Fig. 1(b) is an abstract syntax tree for source file conversion. We can use semantic features to filter candidate values. The parameters defined in the source file or the imported package will be called in the following, such as $(1), (2)$. The source code file imported a Python package $get\_logging$ at position $(1)$, and $get\_logging$ is called at position $(2)$. We extract similar semantic structural features from AST. The defined structure is stored in the collection $vPet$, such as package name or parameters. $vSet$ collection stores the semantic structure of the package or parameter call. Then $vSet - vPet$ is a new candidate value table, which contains the missing nodes. In Fig. 1(c), the candidate value $LOG$ is the OoV word. If the calculated candidate value table is empty, the most frequent $K$ values related to the parent node are selected as candidate values. Finally, the graph embedded model is used to calculate the predicted value, as shown in Fig. 1(d).

The main contribtions of this paper are as follows.

- An effective graph structure is constructed. We extract the path of the terminal node from the AST of the source code and filter out irrelevant paths. These training paths will be converted into training graphs of related nodes. This process can reduce the redundant nodes of the training graphs. The reduced training graphs can speed up the node prediction task.
- The semantic structural features in AST is extracted. We extracted the semantic features can reduce the range of candidate value tables. Since the candidate value is not fixed, it dynamically changes with the prediction file. Thus, the OoV rate can be effectively reduced.
- The efficiency of our model is demonstrated by comparing with the state-of-the-art model. Our model prediction accuracy is improved and the time consumed is shortened. In this model, the graph embedding part can effectively extract the local information of the AST, and the semantic features include the information of global dependence. The combination of these two parts improves the prediction accuracy

of the model. The selection of candidate values by semantic features can shorten the prediction time.

The rest of the paper is organized as follows. The Section 2 introduces related works, and Section 3 explains the basic problem definition. We proposed method in Section 4. Section 5 describes the experiment and the experimental results. Finally, we summarize this paper and outline future work prospects in Section 6.

## 2. Related Works

Simple and effective probability models are widely used in the field of code prediction. The research content is to improve the prediction accuracy of the N-gram language model. Allamanis et al.[3] expanded the size of the corpus based on the research of Hindle et al.[15]. The experiments show that with the increase of data, there is no performance bottleneck in the N-gram model. Nguyen et al.[21] proposed SLAMC to solve the problem that the N-gram model can only extract a limited range of rules, thereby improving the experimental results. Tu et al.[25] found that the locality rules of the code are not used in the N-gram model, and proved that the source code has local repetitiveness. The local repetition rules can be captured by the local cache and applied to software engineering tasks. Nguyen et al.[20] believed that the N-gram model could not extract structural information from API calls, and proposed a language model GraLan for the API call sequence diagram to predict the next API element. In addition to the N-gram model, machine learning methods can also be applied in this field. Bruch et al.[7] used the KNN algorithm to find the most similar completed fragments in the existing code base and provided candidates for method calls. It also points out that many machine learning algorithms can be introduced into the research of code completion. These are similar to the methods recommended for the code, and there are many in life[11][27]. However, these methods have poor ability to deal with OoV problems, and some of them have not considered this issue.

The prediction method based on the representation method of the neural network model mainly uses the improved RNN model, LSTM[18] and gated network[17]. Jian Li[16] proposed parent pointer hybrid network to predict the OoV words in code completion. Adnan Ul et al.[26] used Bi-LSTM model training to split source code identifiers. The model reduces the number of identifier core libraries, thereby improving the accuracy of code prediction. Bhoopchand et al.[6] proposed a new neural network model (sparse pointer network) to capture long-distance dependencies. Context-aware[13] methods have also been used in the Internet of Things. The real-time coding suggestions provided by programmers are closely related to the stability of the network[12][10].

Graph models are widely used in graph structure node quantization processing. Bryan Perozzi et al.[22] proposed that the DeepWalk model uses a random walk algorithm to process static graphs. However, the extraction of DeepWalk path sequences is random, and no attention is paid to the path between Depth-First-Search(DFS) and Breadth First Search(BFS) between fused nodes. Therefore, Aditya Grover et al. [14] proposed Node2vec, which proposed a biased random walk algorithm based on the edge weight of graph nodes. The Node2vec algorithm can fuse the structural information of the node DFS and BFS. Node2vec can effectively integrate the information around the node, so the extracted features are more effective. The powerful method of GNN[9][8] in modeling the dependency relationship between graph nodes has made the research field related to graph analysis achieve good results. Allamanis et al.[2] proposed to use graphs to represent the structure and semantic information of source code, pointing out that graph neural networks have better performance in variable completion and variable misuse than convolutional neural networks, and can complete multiple variables. The latest empirical research by Rahman et al.[23] pointed out that the graph has a higher level of repetitive patterns than the N-gram model. It is recommended to further study the statistical code graph model to accurately capture more complex coding models.

## 3. Problem Definition

In order to facilitate the extraction of the characteristic data in the source code, we converted the source files with different number of lines and volumes into AST. The fixed structure path information of the terminal node is extracted through AST, which is convenient to carry into the downstream model for calculation. All programming languages have clear context-free grammars that can be used to parse source code into AST. Then we extract the path and convert it into a related graph structure (Train-Graph $G$). Finally, the training graph is brought into the model, and the predicted node combination (Node Combination $(T, V)$) is calculated.

**Definition 1:(Train-Graph $G$).** Train-Graph $G = (F, A, Path)$ is a graph converted from the AST's node path of the source file data. $F$ presents a set of $n$ sounce code files, $F$ = $\{f_1, f_2, f_3, ...f_n\}$. $A$ presents a set of Abstract Syntax Tree (AST) which is transformed by context-free grammer $A$ = $\{a_1, a_2, a_3, ...a_n\}$. These AST files contain a majority of node structural feature information. Each $Path$ contains AST's terminal node Up path and Down path. Finally, all node's paths are converted into a training graph $G = (F, A, Path)$.

**Definition 2:(Node Combination $(T, V)$).** $(T, V)$ is calculated by the similarity between the parent node of the prediction node and the candidate value in Train-Graph $G$. $Tc$ represents $s$ candidate values in TYPE task, $Tc$ = $\{Tc_1, Tc_2, Tc_3, ...Tc_s\}$. $Vc$ represents $k$ candidate values in VALUE task, $Vc$ = $\{Vc_1, Vc_2, Vc_3, ...Vc_k\}$. $T$ is the maximum value calculated by predicting the similarity between the parent node $p\_node$ and $Tc$, $T = S(G, Tc, p\_node)$. $V$ is the maximum value calculated by predicting the similarity between the parent node $T$ and $V$, $V = S(G, Vc, T)$.

For example, as shown in Fig. 1, we predict the missing node=$(T, V)$ in the last line of code. In the corresponding AST, it can be obtained that the $p\_node$ is $Assign$ in Fig. 1(b). After embedding the graph node, the model will calculate the vector similarity between the candidate value and the parent node($Assign$) of the predicted node. If the candidate node structure is more similar to the parent node, the more likely the two are connected. So the TYPE prediction task is to calculate $T = S(G, Tc, Assign)$. Determine the type of the prediction node, then we use the type as the new parent node and calculate $V = S(G, Vc, T)$ to predict the VALUE task. This prediction process is transformed into the prediction of the node, as shown in the following Equation (1), (2).

$$\exists i \in \{1, 2, 3, ...s\} : T = S(G, Tc, p\_node) \geq S(G, Tc_i, p\_node) \tag{1}$$

$$\exists j \in \{1, 2, 3, ...k\} : V = S(G, Vc, p\_node) \geq S(G, Vc_j, p\_node) \tag{2}$$

Therefore, the solution is to calculate the similarity between each candidate value and the parent node vector of the predicted node. Finally, we obtain the prediction token $(T, V)$.

# 4. The Proposed Approach

This Section contains four parts: The Framework, Data Processing, Semantic Feature Extraction and Graph Embedding Mode.

## 4.1. The Framework

Fig. 2 shows the main architecture of our proposed model. The overall framework of this paper is mainly divided into three parts: Data Processing, Semantic Feature Extraction, and Graph Embedding Model.

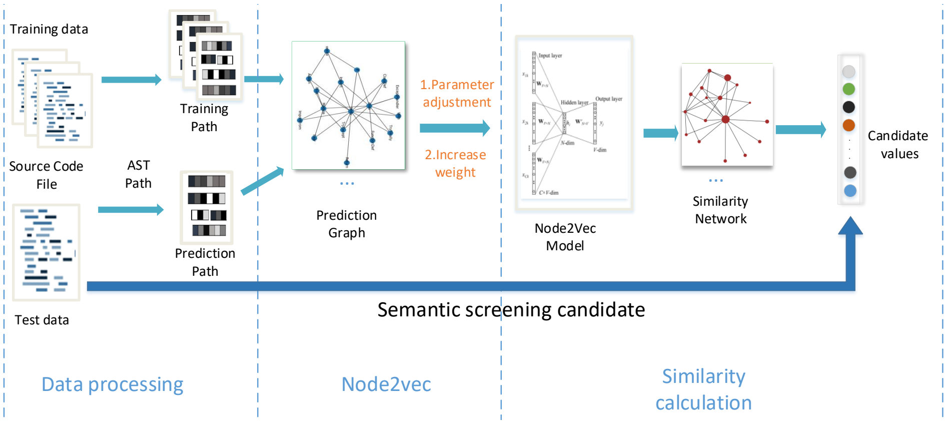
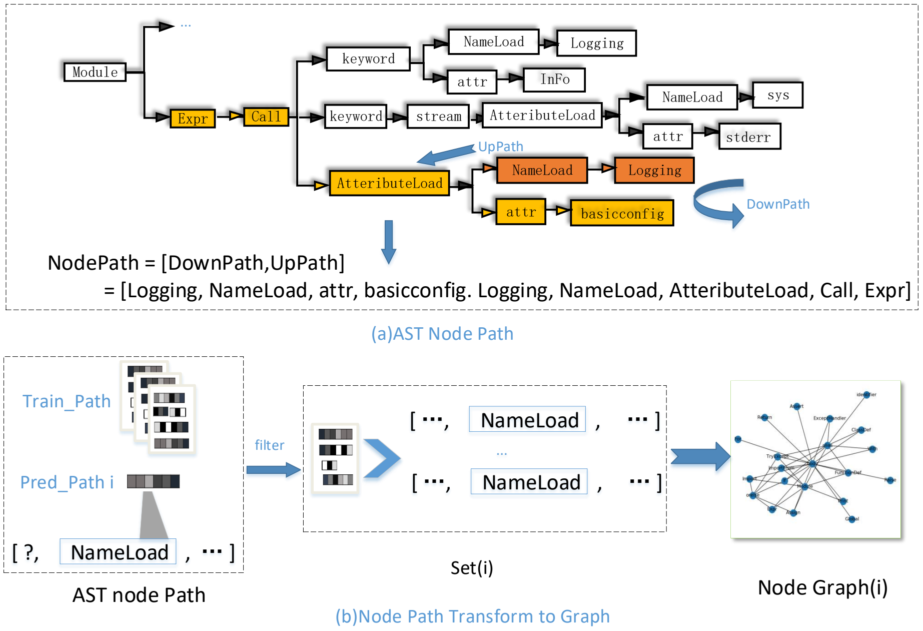


**Fig. 2.** The model framework

The data processing part uses context-free grammar to transform the source file into an abstract syntax tree. We extract the path of the AST terminal node and convert it into a training graph. Semantic structure feature can extract global information of remote dependencies, and the extracted semantic structure can filter out new candidate values. The graph embedding model uses the Node2vec algorithm to reduce the dimensionality of the node graph. It biasly extracts the relevant node sequences and converts them into vector of fixed dimensions, and the vector will bring into the downstream prediction task for calculation.

## 4.2. Data Processing

After converting the source code to AST, we extract the path of the terminal node. As shown in Fig. 3(a), take the terminal node *logging* as an example. We obtain its upward path $UpPath$ and downward path $DownPath$, and merge the two path to get the $logging$ node's path $NodePath$. $UpPath$ contains the node hierarchy information of AST, which can reflect the hierarchical characteristics of nodes in AST. $DownPath$ mainly extracts local information of terminal nodes and can directly reflect the correlation of neighboring nodes. Each node path contains rich and steric information.



(a)AST Node Path

(b)Node Path Transform to Graph

**Fig. 3.** Data processing

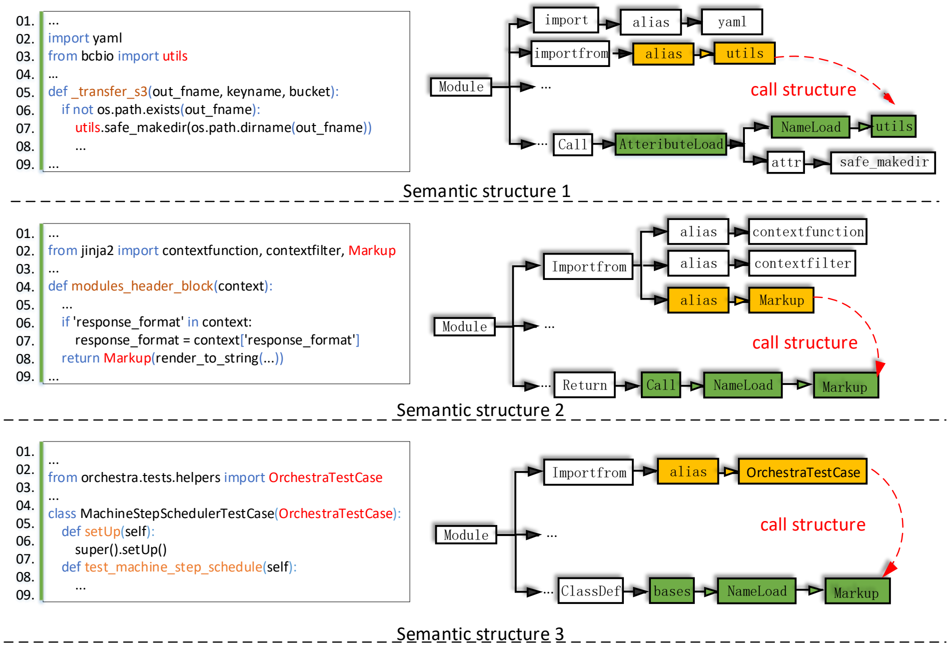
For each prediction path, we make full use of the existing structural information. The parent node of the terminal node can provide information intuitively, so we use the parent node to filter the training data set. In particular, each prediction data will have a corresponding data set and generate a corresponding graph structure. As shown in Fig. 3(b), the known parent node $NameLoad$ of the predicted path is used to filter out the data set $Set(i)$ and convert it into the corresponding graph $G(i)$. For edges extracted from the source data, the model will count the number of occurrences and use this as the weight of the graph $G(i)$.

After data processing, we can remove a lot of redundant information in the source code files. This process not only reduces the node size of the graph, but also retains useful information.

### 4.3. Semantic Feature Extraction

Programming language is a natural language with obvious repetitive characteristics[1]. We can extract the semantic features of the programming language from the contextual relevance to help us complete specific tasks. In this paper, we analyze the structure of the AST of the source file and combine the code semantics to find out the rules between the structures. In traditional prediction tasks, the $Top - K$ values with the most occurrences in the training data are established as candidate values (Section 4.1). But there are two conflicting issues in this process:

- The range of the candidate value table is small. Since the number of candidate values is small, there is a high probability that the predicted node value is not in the candidate value table. This situation will not only cause OoV problems, but also reduce the prediction accuracy.
- The candidate value table has a larger range. Since there are too many candidate values, the prediction accuracy will be improved. However, each prediction process must calculate a large number of irrelevant candidate nodes. This leads to a significant increase in the prediction time of the model.



```
...
import yaml
from bcbio import utils
...
def _transfer_s3(out_fname, keyname, bucket):
    if not os.path.exists(out_fname):
        utils.safe_makedir(os.path.dirname(out_fname))
        ...
...
```

```
...
from jinja2 import contextfunction, contextfilter, Markup
...
def modules_header_block(context):
    ...
    if 'response_format' in context:
        response_format = context['response_format']
    return Markup(render_to_string(...))
...
```

```
...
from orchestra.tests.helpers import OrchestraTestCase
...
class MachineStepSchedulerTestCase(OrchestraTestCase):
    def setUp(self):
        super().setUp()
    def test_machine_step_schedule(self):
        ...
```

**Fig. 4.** The semantic structure of calling python packages

Therefore, the core problem of the prediction model is to determine a valid candidate value table. In neural network models, researchers often choose tens of thousands of fixed candidate values. Compared with the traditional model, although the accuracy of the neural network model has been improved, it consumes longer time. In order to solve the problem, this paper dynamically selects candidate value tables by extracting contextual semantic features.

This method is mainly to solve the VALUE prediction task of the missing node. Because there are only about 100 candidate values in the prediction task of TYPE, these words are determined by the programming language. However, most node values are defined by programmers. The randomness of these values is large, which leads to a wide range of candidate values. As shown in the Fig. 4, take the name of the missing package in python as an example:

In the programming language, the packages to be called need to be defined in advance (AST structure: $alias \to value$). We found that the code called and imported has a fixed semantic structure in AST. The called package has six structures, but the most important are the three semantic structures shown in Fig. 4. We record the known package name in the $vPet$ collection, and record the called package in the $vSet$ collection. $vSet - vPet$ is a candidate value for the called but missing defined package name. Conversely, if the missing node is the code that calls the package, $vPet - vSet$ will be the new candidate value table. When $vSet - vPet$ is empty or the predicted value is not called below, we select the $K$ most frequently occurring values related to the parent node as candidate values. This process can remove the candidate words that are completely impossible to predict. Similarly, we can use the parent node semantics of the prediction node to extract other semantic structural features, such as $NameLoad \to value$, $NameStore \to value$, etc. By extracting the context semantic structure of the terminal node, we can reduce thousands of candidate values to tens or even a few. Since the obtained candidate value changes with the prediction file, it can effectively reduce the OoV rate.

The pseudo code for semantic feature extraction algorithm is as shown in:

**Algorithm 1:** Semantic Feature Extraction Algorithm

```
Input: Prediction File $PF$, number $n$;
       Prediction Data $TeD$ = $(DownPath, UpPath)$;
       Candidate value $Candidate\_Value$ ;
Output: New Candidate Value: $New\_Candidate\_Value$ ;
$p\_node \leftarrow$ Each $TeD's$ parent node
for $i = 1 : n$ do
    for $j = 1 : len(PF_i)$ do
        if $Parent\_node ==' alias'$ then
            $vSet.append(Structure(AtteributeLoad \to NameLoad \to Value))$;
            $vSet.append(Structure(Call \to NameLoad \to Value))$;
            $vSet.append(Structure(bases \to NameLoad \to Value))$;
            $vPet.append(Structure(alias \to Value))$;
        end
        $PF_i$ $New\_Candidate\_Value$ = $vSet - vPet$;
        Similarly, filter other semantic features by different $p\_node$.
    end
    if $PF_i$ $New\_Candidate\_Value$ == None then
        $PF_i$ $New\_Candidate\_Value$ = $Candidate\_Value$;
    end
end
```

### 4.4. Graph Embedding Model

The sequence of graph nodes in the DeepWalk algorithm is randomly extracted, while the sequence extraction of Node2vec combines DFS and BFS of nodes. Node2vec is a graph embedding method that comprehensively considers the DFS neighborhood and BFS neighborhood information of the graph. It is regarded as an extension algorithm of DeepWalk.

**Random Walk :** Node2vec obtains the neighbor sequence of vertices in the graph by biased random walk, which is different from DeepWalk. Given the current vertex $v$, the probability of visiting the next vertex $x$ is:

$$P(c_i = x|c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} \ if(v,x) \in E \\ 0 \quad otherwise \end{cases} \tag{3}$$

where $\pi_{vx}$ is the unnormalized transition probability between nodes $v$ and $x$, and $Z$ is the normalizing constant.

**Search Bias $\alpha$ :** The simplest method of biased random walk is to sample the next node according to the weight of the edge. However, this method does not allow us to adjust the search process to explore different types of network neighbors. Therefore, the biased random walk should be a fusion of DFS and BFS, rather than mutually exclusive. The model should combine the structural features and content features between the nodes.

The two parameters $p$ and $q$ which guide the random walk. As shown in Fig. 5, we suppose that the current random walk through the edge $(t, v)$ reaches the vertex $v$, edge labels indicate search biases $\alpha$. The walk path now needs to decide on the next step. The method will evaluate the transition probabilities $\pi_{vx}$ on edges $(v, x)$ leading from $v$. Node2vec set the transition probability to $\pi_{vx} = \alpha_{pq}(t, x) * w_{vx}$, where

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} \ if \quad d_{tx} = 0 \\ 1 \ if \quad d_{tx} = 1 \\ \frac{1}{q} \ if \quad d_{tx} = 2 \end{cases} \tag{4}$$

and $w_{vx}$ is the edge weight between nodes.

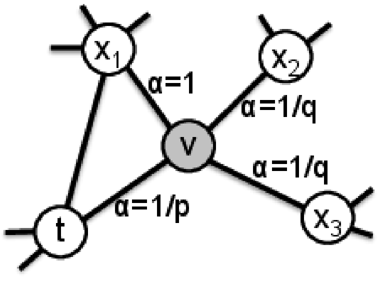


**Fig. 5.** The next step out of node v

**Return parameter, p:**

- **If** $p > max(q, 1)$ The probability of sampling to the original node is very small. As shown in Fig. 5, the probability of the next node returning to node $t$ is low.
- **If** $p < max(q, 1)$ The probability of node sampling returning to the previous node is high. This causes some node paths to move around the starting point.

**In-out papameter, q:**

- **If** $q > 1$ The node sequence will move among the nodes near the starting point, which can reflect the BFS characteristics of the node.
- **If** $q < 1$ The node sequence will become farther from the starting node, and the return probability is small. This reflects the characteristics of DFS.

When $p = 1$ and $q = 1$, the walk mode is equivalent to the random walk in DeepWalk.

**Similarity Calculation :** The bias random walk algorithm can extract the node structure path in train graph. Then, the model use Word2vec's SkipGram algorithm to convert the node sequence into a vector of fixed dimensions. The Word2vec model can reduce the dimensionality of discrete text data into quantifiable vectors. We use the cosine function to calculate the structural coincidence between nodes. The candidate node with the largest cosine value also contains the most of the same node structure feature.

The calculation function of the similarity between nodes is shown in Equation 5:

$$cos(\theta) = \frac{V_1 \cdot V_2}{\|V_1\| \, \|V_2\|} = \frac{\sum_{i=1}^{n} V_{1i} \times V_{2i}}{\sqrt{\sum_{i=1}^{n} (V_{1i})^2} \times \sqrt{\sum_{i=1}^{n} (V_{2i})^2}} \tag{5}$$

The pseudo code for our entire model: **Algorithm 2**

**Algorithm 2:** Graph Embedding Code Prediction

```
Input: Train Data $TD$=($DownPath, UpPath$),number $n$;
       Prediction Data $TeD$ = ($DownPath, UpPath$);
       New Canditite Value $New\_Canditite\_Value$ and number $s$ ;
Output: Predicted Value $pred\_value$ ;
$p\_node \leftarrow$ Each $TeD's$ parent node
for $i = 1 : n$ do
    $Set \leftarrow$ p_node in $TD_i$
end
$G(i) \leftarrow$ Produce graph by Set's node
$G'(i) \leftarrow$ Increase the $G(i)$ weight of edges related to $Ted$
$Embed\_model \leftarrow$ Node2vec($G'(i), p, q$), adjusting parameters $p$, $q$
for $j = 1 : s$ do
    $V1 = Embed\_model(p\_node)$
    $V2 = Embed\_model(Cand\_Value_j)$
    $SimScore \leftarrow Cos(V1, V2)$
end
$pred\_value \leftarrow Max(SimScore)$
```

## 5. Experiment Set Up

The experiment is mainly divided into four parts. First of all, we introduce the data set. Then, we discuss the prediction tasks of TYPE and VALUE. In the last part, the paper discusses the experimental results. The experimental hardware environment is Intel(R) Xeon(R) CPU E5-2660 v4 @ 2.00GHz; RAM 32.00GB.

### 5.1. Data Set

In the experiment, we collect Python data set from the Github[4] repository. The data set contains 10,000 training data files and 500 prediction data files. These Python source files have high star mark in github and are public available.

As shown in Table 1, In the TYPE prediction task, there are only 132 types. Such as $NameLoad$, $alias$, $NameParam$, etc. These types are determined by the Python programming language and cannot be set by the programmer, which results in fewer candidate values. In the VALUE prediction task, the source file has 51,000 different node values. There are arbitrary possibilities for encoding the program text. The value can be any program identifier (such as $None$, $format$), literals (such as $0.035$, $1075$), program operators (such as $/$, $-$, $*$), etc. It is impossible to use all of them for calculation, especially some of these values only appear once, so we need to filter the vocabulary.

**Table 1.** Dataset statistics

| | Category | Size |
|---|---|---|
| 1 | Type Vocabulary | 132 |
| 2 | Value Vocabulary | $5.1*10^5$ |
| 3 | Training files | 10000 |
| 4 | Test files | 500 |

**Table 2.** TYPE Nodes type

| | Types | Size |
|---|---|---|
| 1 | NameLoad | $1.2*10^6$ |
| 2 | attr | $1.1*10^6$ |
| 3 | AttributeLoad | $8.4*10^5$ |
| 4 | Str | $5.1*10^5$ |
| ... | ... | ... |
| 132 | CompareLtELtELtE | 1 |

**Table 3.** VALUE Nodes type

| | Types | Size |
|---|---|---|
| 1 | self | $2.8*10^5$ |
| 2 | None | $4.0*10^4$ |
| 3 | 0 | $3.8*10^4$ |
| 4 | 1 | $3.4*10^4$ |
| ... | ... | ... |
| $5.1*10^5$ | Sysbench-read cleanup on %s | 1 |

[4] https://github.com

We extract all the node types in the source code files, a total of 132 different types. As shown in Table 2, we can find that there is a significant difference between the maximum and minimum number of occurrences of node's type. For example, $CompareLtELtELtE$ appears only once in the source data, and it has little effect on other tens of thousands of candidate values during the experiment. Therefore, we choose the type value that appears more than 100 times as the candidate value in the prediction TYPE task.

As shown in Table 3, the number of node values in the training code source file is $5.1 * 10^5$, and most of them are random strings defined by programmers. The number of unique node values in dataset is too large to directly apply neural networks models and the first type $self$ is 2.8*$10^5$ times different from the last value $Sysbench - read\ cleanup\ on\ \%s$, thus we choose $K = 1000$ most frequent values in all training set to build the global vocabulary of type.

However, using the $Top - K$ values in the training data as fixed candidate values, this method cannot avoid OoV problems. Therefore, we need to dynamically select candidate values with semantic structural features.

### 5.2. TYPE Prediction

In the TYPE prediction task, the number of candidate values is small, and none are randomly defined by the programmer. Therefore, the model mainly learns the structural features of the nodes in the training graph and completes the prediction of the node types. Secondly, the candidate values of type are fixed and will not change with different prediction data. This is the biggest difference from the prediction task of VALUE.

The selection of the candidate value of TYPE is mainly filtered by the parent node of the terminal node through two methods:

- **Semantic Extraction:** For a fixed code structure, we extract the specific AST path structure. For example, when importing a package, $Import \rightarrow alias$ is a fixed structure, and $Import$ does not appear at other AST node locations.
- **Traverse the Dataset:** Traverse the parent node of each terminal node of all training data, this method is suitable for TYPE prediction tasks. Because the number of TYPEs is limited, the short traversal time can accurately filter out candidate values.

The graph embedding model mainly extracts related node sequences between graph nodes, and uses Word2vec[19] to convert these sequences into fixed-dimensional vectors. Then, we calculate the vector similarity between the candidate value and the parent node of the prediction node. The higher the similarity between nodes, the greater the degree of coincidence of the path structure extracted by the graph model. Therefore, the probability of connection between nodes is higher. In the extraction of graph node sequence, it is mainly affected by the parameters $p$ and $q$. The parameters $p$ and $q$ affect the search method of the node sequence, so the information contained in the node sequence is different. In order to distinguish different prediction paths with the same parent node, we increase the weight of known nodes in the prediction path. This method can effectively distinguish the paths with the same parent node but different prediction values. After obtaining the training graph related to the predicted data, we train the model to determine the parameter $(p, q)$. By changing the range of the parameters $(p, q)$ with the model accuracy, the approximate range is determined. For example $p > 1, q < 1$, then we fine-tune the model to determine the exact values of the parameters.

**Table 4.** Candidates for TYPE

| **Father Node** | **Candidate values** | **(p,q) value** |
|---|---|---|
| ImportFrom | alias | (p=1.8, q=0.9) |
| Import | alias | (p=2, q=0.8) |
| args | NameParam, TupleStore | (p=2, q=0.3) |
| alias | identifier | (p=1.8, q=0.8) |
| bases | NameLoad, AttributeLoad, Call | (p=2, q=0.8) |
| AttributeLoad | NameLoad, attr, Str, Num | (p=1.7, q=0.8) |
| keyword | NameLoad, attr, Str, Num | (p=0.8, q=1.5) |
| Assign | NameStore, Call, ListLoad, NameLoad, Str, IfExp, GeneratorExp, BinOpMod, SubscriptLoad, AttributeLoad, BoolOpOr, ... | (p=1.6, q=0.5) |
| Call | NameLoad, AttributeLoad, TupleLoad, keyword, ListLoad, BinOpMod, Str, Dict, Num, ListComp, BinOpMult, ... | (p=1.5, q=0.5) |
| TupleLoad | NameLoad, AttributeLoad, Str, SubscriptLoad, BinOpAdd, Num, ListLoad, Call, BinOpSub, Dict... | (p=1.5, q=0.9) |
| ListLoad | NameLoad, Str, Num, Call, AttributeLoad, BinOpAdd, TupleLoad, ListLoad, Dict, BinOpMod,... | (p=2, q=0.8) |
| ... | ... | ... |

In Table 4, there are only a few candidate values for $Import$, $importForm$, and $Assign$ has a maximum of 50 candidate values. So we can see that the range of candidate values has been significantly reduced, and the candidate values of TYPE are fixed after screening. This process can not only increase the accuracy of prediction, but also shorten the prediction time. In model prediction, the model mainly extracts node sequences based on depth-first search$(p > 1, q < 1)$. However, the node sequence of breadth-first search is also included, such as keywords, parameters $(p = 0.8, q = 1.5)$.

### 5.3. Value Prediction

The prediction task of VALUE is much more difficult than the prediction task of TYPE. First of all, the candidate value of can reach tens of thousands in VALUE prediction task. Secondly, for the artificially defined word names of programmers, it is difficult to obtain effective structural feature information in training data. Especially for nodes of type $Str$, the range of candidate values is very large and random. But during the experiment, we can extract the candidate values of the semantic structure of the prediction file. We extract the semantic features of the AST according to the parent node of the prediction node and filter out new candidate values. Such dynamic candidate values can effectively solve the OoV problem.

Similar to the TYPE prediction task, we mainly use the graph embedding model to extract related node sequences between graph nodes, and Word2vec algorithm converts these sequences into fixed-dimensional vector.

For the parent nodes with obvious structural semantics, such as $NameLoad$, $alias$, $NameParam$ and $NameStore$, we use the semantic structure features to filter them. As shown in Table 5, the node sequence extraction of the model is based on DFS.

**Table 5.** Semantic structure for VALUE

| Father Node | Related Semantic Structure | (p,q) Value |
|---|---|---|
| alias | $identifier/NameStore/alias... \rightarrow value$, $Call/bases/AttributeLoad... \rightarrow NameLoad \rightarrow value$ | (p=2, q=0.8) |
| NameLoad | $alias/NameParam/FunctionDef... \rightarrow value$, $bases/Raise... \rightarrow NameLoad \rightarrow value$ | (p=2, q=0.8) |
| NameStore | $identifier/NameParam/alias... \rightarrow value$, $Call/AttributeLoad... \rightarrow NameLoad \rightarrow value$ | (p=1.6, q=0.5) |
| NameParam | $NameStore/NameParam/alias... \rightarrow value$, $Call/AttributeLoad... \rightarrow NameLoad \rightarrow value$ | (p=2, q=0.3) |
| ... | ... | ... |

But for nodes that are randomly defined by the programmer and will not be called below. We traverse all training data sets and select the $Top - K$ words that appear most frequently as related parent nodes as candidate values. Compared with the traditional method of directly extracting the most frequent values. The advantage of this method is that the parent node can select candidate values to remove redundant words. For example, if the parent node is $Num$, We will extract the numbers in the training data as candidate values. And the most frequently occurring $self$ will not be considered as a candidate. But in the traditional prediction model, $self$ will be brought into the model for calculation.

### 5.4. Experimental Results

First of all, we introduce prediction accuracy to evaluate the performance of our proposed model, which can be described as Eq.(6):

$$Accuracy = \frac{The\ number\ of\ correct\ prediction\ nodes}{The\ total\ number\ of\ prediction\ nodes} \tag{6}$$

The experimental results compared to the state-of-the-art[16] model in the same data set are shown in the table below:

As can be seen from Table 6, compared with the state-of-the-art model experimental results, our model has better results. Especially in the prediction task of TYPE, its accuracy has improved significantly. The main reason are: 1. We screened the candidate values of TYPE, narrowing the candidate range from hundreds to dozens or even a few. 2. The main prediction of TYPE is the structure of the code, and the path and training map we extract are highly relevant to the code structure. 3. There are few node types in the TYPE task, so the network scope of the composed node graph is small, resulting in improved prediction accuracy. In other words, the model effectively extracts the structural

**Table 6.** Comparison of final results

| | TYPE | VALUE |
|---|---|---|
| **Attentional LSTM** | 71.1% | - |
| **Pointer Mixture Network** | - | 62.2% |
| **Our Model** | **77.8%** | **63.8%** |

features of the code. In the prediction task of VALUE, the accuracy is improved by 1.6%. Although the accuracy has not improved much, the overall prediction time of the model has been reduced significantly.

**Table 7.** Comparison of prediction time cost

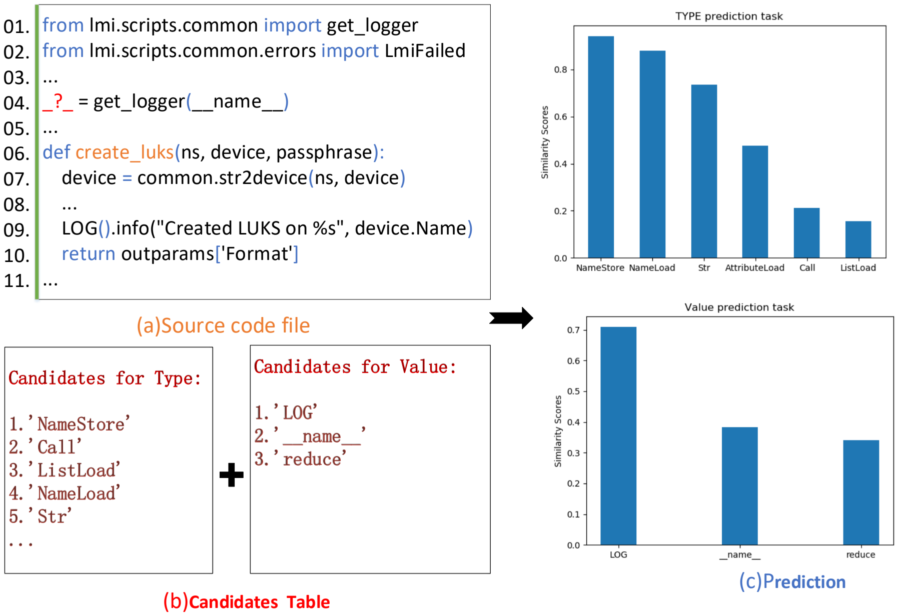
| | Total Time | Type Task Time | Value Task Time |
|---|---|---|---|
| **Attentional LSTM** | >20h | - | - |
| **Pointer Mixture Network** | | | |
| **Our Model** | 8.5h | 2.3h | 6.2h |

As shown in Table 7, the running time of the deep learning model method on the same data set exceeds 20 hours. And the interpretability of the deep learning model is low. The total consumption time of our proposed model is 8.5 hours, and the complex VALUE prediction task takes about 70% of the time. This is mainly because the training graph has more nodes than the training graph of the TYPE task, so the embedding of the graph takes longer time.

The model graph structure feature extraction proposed in this paper is more interpretable and more intuitive than deep learning model for prediction tasks. In the overall graph embedded in the model, the value range of the $(p, q)$ parameter is mainly $(p > 1, q < 1)$. The node sequence of the model incorporates more depth-first node information.

We predict the example in Fig. 1 and the result is shown in Fig. 6. We calculated the similarity between the node in the graph structure and the parent node $Assign$. After model calculation, we find that the type of the missing node is $NameStore$. Then we get the candidate value table through the screening of the code semantic structure, and calculate the predicted value of the OoV word $LOG$ as the final value. The missing node is $(T, V) = (NameLoad, LOG)$. Experiments show that the OoV rate in the predicted data drops from 19.6% to 5.7%, and most of them are $Str$ nodes that fail to accurately predict. The overall OoV rate has dropped significantly, and the prediction accuracy has also been significantly improved.

However, nodes of type $Str$ are difficult to predict. First of all, these words may have been created by the programmer themselves, so they will basically not appear in the candidate value table. Secondly, these words appear rarely. they do not even appear in



```
from lmi.scripts.common import get_logger
from lmi.scripts.common.errors import LmiFailed
...
_?_ = get_logger(__name__)
...
def create_luks(ns, device, passphrase):
    device = common.str2device(ns, device)
    ...
    LOG().info("Created LUKS on %s", device.Name)
    return outparams['Format']
...
```

(a)Source code file

Candidates for Type:

1.'NameStore'
2.'Call'
3.'ListLoad'
4.'NameLoad'
5.'Str'
...

Candidates for Value:

1.'LOG'
2.'__name__'
3.'reduce'

(b)Candidates Table

(c)Prediction

**Fig. 6.** Code prediction example

the training Graph, which leads to a reduction in the overall prediction accuracy of value. This is also a disadvantage of this model.

## 6. Conclusion and Future Work

In this paper, we use the node information of the source code AST to construct training graphs, which contain a lot of node structure information. Through the embedding model, we can embed the graph structure node sequence as a fixed-dimensional vector. Then carry it to the downstream task for calculation. In the selection of candidate values, we use semantic structural features to dynamically filter candidate values, which not only reduces the prediction time, but also effectively reduces the OoV situation.

The experimental results show that the model can effectively extract structural features in the prediction task of TYPE, and the prediction accuracy is greatly improved. In the prediction task of VALUE, the screening of candidate values not only improves the accuracy, but also shortens the time.

The accuracy improvement of TYPE shows that the accuracy of the programming language with more strict structure will be more obvious. Therefore, in future work we will add Java data sets to verify our model. And, we will further extract the semantic features of the fusion Control Flow Graph(CFG).

**Acknowledgments.** This work is partially supported by the NSF of China under grants No.61702334 and No.61772200, the Project Supported by Shanghai Natural Science Foundation No.17ZR 1406900, 17ZR1429700 and Planning project of Shanghai Institute of Higher Education No.GJEL18135.

## References

1. Allamanis, M., Barr, E.T., Devanbu, P.T., Sutton, C.A.: A survey of machine learning for big code and naturalness. ACM Comput. Surv. 51(4), 81:1–81:37 (2018)
2. Allamanis, M., Brockschmidt, M., Khademi, M.: Learning to represent programs with graphs. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings (2018)
3. Allamanis, M., Sutton, C.A.: Mining source code repositories at massive scale using language modeling. In: Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13, San Francisco, CA, USA, May 18-19, 2013. pp. 207–216 (2013)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
5. Bahl, L.R., Jelinek, F., Mercer, R.L.: A maximum likelihood approach to continuous speech recognition. IEEE Trans. Pattern Anal. Mach. Intell. 5(2), 179–190 (1983)
6. Bhoopchand, A., Rocktäschel, T., Barr, E.T., Riedel, S.: Learning python code suggestion with a sparse pointer network. CoRR abs/1611.08307 (2016)
7. Bruch, M., Monperrus, M., Mezini, M.: Learning from examples to improve code completion systems. In: Proceedings of the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering, 2009, Amsterdam, The Netherlands, August 24-28, 2009. pp. 213–222 (2009)
8. Cao, S., Lu, W., Xu, Q.: Deep neural networks for learning graph representations. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA. pp. 1145–1152 (2016)
9. Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada. pp. 2224–2232 (2015)
10. Gao, H., Huang, W., Duan., Y.: The cloud-edge based dynamic reconfiguration to service workflow for mobile ecommerce environments: A qos prediction perspective. ACM Transactions on Internet Technology (2020)
11. Gao, H., Kuang, L., Yin, Y., Guo, B., Dou., K.: Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps. ACM/Springer Mobile Networks and Applications (MONET) (2020)

12. Gao, H., Liu, C., Li, Y., Yang., X.: V2vr: Reliable hybrid-network-oriented v2v data transmission and routing considering rsus and connectivity probability. IEEE Transactions on Intelligent Transportation Systems pp. 1–14 (2020)
13. Gao, H., Xu, Y., Yin, Y., Zhang, W., Li, R., Wang., X.: Context-aware qos prediction with neural collaborative filtering for internet-of-things services. IEEE Internet of Things Journal 7(5), 4532–4542 (2020)
14. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. pp. 855–864 (2016)
15. Hindle, A., Barr, E.T., Gabel, M., Su, Z., Devanbu, P.T.: On the naturalness of software. Commun. ACM 59(5), 122–131
16. Li, J., Wang, Y., Lyu, M.R., King, I.: Code completion with neural attention and pointer networks. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden. pp. 4159–4165 (2018)
17. Li, Y., Tarlow, D., Brockschmidt, M., Zemel, R.S.: Gated graph sequence neural networks. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings (2016)
18. Malhotra, P., Vig, L., Shroff, G.M., Agarwal, P.: Long short term memory networks for anomaly detection in time series. In: 23rd European Symposium on Artificial Neural Networks, ESANN 2015, Bruges, Belgium, April 22-24, 2015 (2015)
19. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings (2013)
20. Nguyen, A.T., Hilton, M., Codoban, M., Nguyen, H.A., Mast, L., Rademacher, E., Nguyen, T.N., Dig, D.: API code recommendation using statistical learning from fine-grained changes. In: Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2016, Seattle, WA, USA, November 13-18, 2016. pp. 511–522 (2016)
21. Nguyen, T.T., Nguyen, A.T., Nguyen, H.A., Nguyen, T.N.: A statistical semantic language model for source code. In: Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE'13, Saint Petersburg, Russian Federation, August 18-26, 2013. pp. 532–542 (2013)
22. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations. In: The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014. pp. 701–710 (2014)
23. Rahman, M., Palani, D., Rigby, P.C.: Natural software revisited. In: Proceedings of the 41st International Conference on Software Engineering, ICSE 2019, Montreal, QC, Canada, May 25-31, 2019. pp. 37–48 (2019)
24. Raychev, V., Vechev, M.T., Yahav, E.: Code completion with statistical language models. In: ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI '14, Edinburgh, United Kingdom - June 09 - 11, 2014. pp. 419–428 (2014)

25. Tu, Z., Su, Z., Devanbu, P.T.: On the localness of software. In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, (FSE-22), Hong Kong, China, November 16 - 22, 2014. pp. 269–280 (2014)
26. Ul-Hasan, A., Ahmed, S.B., Rashid, S.F., Shafait, F., Breuel, T.M.: Offline printed urdu nastaleeq script recognition with bidirectional LSTM networks. In: 12th International Conference on Document Analysis and Recognition, ICDAR 2013, Washington, DC, USA, August 25-28, 2013. pp. 1061–1065 (2013)
27. Yang, X., Zhou, S., Cao, M.: An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: The product-attribute perspective from user reviews. MONET 25(2), 376–390 (2020)

**Kang Yang** received his B.S. degree from Shanghai Ocean University in 2017. He is a Ph.D. student in computer science at East China University of Science and Technology. His research interests include software engineering, natural language processing.

**Huiqun Yu** received his B.S. degree from Nanjing University in 1989, M.S. degree from East China University of Science and Technology (ECUST) in 1992, and Ph.D. degree from Shanghai Jiaotong University in 1995, all in computer science. He is currently a Professor of computer science with the Department of Computer Science and Engineering at ECUST. From 2001 to 2004, he was a Visiting Researcher in the School of Computer Science at Florida International University. His research interests include software engineering, high confidence computing systems, cloud computing and formal methods. He is a member of the ACM, a senior member of the IEEE, and a senior member of the China Computer Federation.

**Guisheng Fan** received his B.S. degree from Anhui University of Technology in 2003, M.S.degree from East China University of Science and Technology (ECUST) in 2006, and Ph.D. degree from East China University of Science and Technology in 2009, all in computer science. He is presently a research assistant of the Department of Computer Science and Engineering, East China University of Science and Technology. His research interests include formal methods for complex software system, service oriented computing, and techniques for analysis of software architecture.

**Xingguang Yang** received his B.S. degree from China University of Mining and Technology in 2016. He is a Ph.D. student in computer science at East China University of Science and Technology. His research interests include software engineering, Software defect prediction.

*Received: September 8, 2019; Accepted: July 7, 2020.*

Computer Science and Information Systems 17(3):927–958 https://doi.org/10.2298/CSIS200127028S

# Improving the Performance of Process Discovery Algorithms by Instance Selection*

Mohammadreza Fani Sani[1], Sebastiaan J. van Zelst[1,2], and Wil van der Aalst[1,2]

[1] Process and Data Science Chair, RWTH Aachen University, Aachen, Germany
{fanisani, s.j.v.zelst, wvdaalst}@pads.rwth-aachen.de
[2] Fraunhofer FIT, Birlinghoven Castle, Sankt Augustin, Germany

**Abstract.** Process discovery algorithms automatically discover process models based on event data that is captured during the execution of business processes. These algorithms tend to use all of the event data to discover a process model. When dealing with large event logs, it is no longer feasible using standard hardware in limited time. A straightforward approach to overcome this problem is to down-size the event data by means of sampling. However, little research has been conducted on selecting the right sample, given the available time and characteristics of event data. This paper evaluates various subset selection methods and evaluates their performance on real event data. The proposed methods have been implemented in both the ProM and the RapidProM platforms. Our experiments show that it is possible to considerably speed up discovery using instance selection strategies. Furthermore, results show that applying biased selection of the process instances compared to random sampling will result in simpler process models with higher quality.

**Keywords:** Process Mining, Process Discovery, Subset Selection, Event Log Preprocessing, Performance Enhancement.

## 1. Introduction

*Process Mining* bridges the gap between traditional data mining and business process management analysis[1]. The main subfields of process mining are 1) *process discovery*, i.e., finding a descriptive model of the underlying process, 2) *conformance checking*, i.e, monitoring and inspecting whether the execution of the process in reality conforms to the corresponding designed (or discovered) reference process model, and 3) *enhancement*, i.e, the improvement of a process model, based on the related event data [1]. With process discovery, we aim to discover a process model that accurately describes the underlying process captured within the event data, also referred to as *event logs*, readily available in most modern information systems. In conformance checking, we aim to assess to what degree a given process model (possibly the result of a process discovery algorithm) and event data conform to one another. Finally, process enhancement aims at improving the view on a process by improving or enhancing the corresponding model using related event data, e.g., by projecting bottleneck information directly onto a (given) process model. There are also other dimensions like prediction[39, 38] and business process automation [24].

* This article is the extension of a conference paper entitled ”The Impact of Event Log Subset Selection on the Performance of Process Discovery Algorithms” that was initially published in 23rd European Conference on Advances in Databases and Information Systems 2019 Workshops proceedings (Springer CCIS 1064) [41].

Currently, the main research focus of process discovery is on quality issues of the discovered process models; however, at the same time, the ever-increasing size of the data handled in process mining leads to performance issues when applying the existing process discovery algorithms [2]. Most process discovery algorithms first build an internal abstraction of data, based on the whole event log and then apply a possible filtering step is applied. However, this attitude causes efficiency in real process mining projects with large event data. Some process discovery algorithms are infeasible in big data settings, where the event data are too large to process. Additionally, some process mining tools enforce constraints on the size of event data, e.g., the number of events. Also, in many cases, we do not require the whole event log, and an approximation of the process model is able to be discovered by applying just a small fraction of the event data.

In real life, process discovery is often of an exploratory nature which means sometimes we need to apply different process discovery algorithms with several parameters to generate different process models and select the most suitable process model. When the discovery algorithms are used repeatedly, such an exploratory approach makes sense only if performance is reasonable. Thus, even a small improvement in performance may accumulate to a significant performance increase when applied several times. Furthermore, many process discovery algorithms are designed to also generalize the behavior that is observed in the event data. In other words, the result of process discovery algorithms contains more behavior compared to the inputted event log. Therefore, it may still be possible to discover the underlying process using a subset of event data. In addition, many of process discovery algorithms aim to depict as much as possible behavior in event logs. However, for real event logs, because of the presence of noisy and uncertain behavior, applying these algorithms resulted in inaccurate and incomprehensible, complex process models that are even accurate behavior undetectable in them [18, 36].

To deal with the complexity of process models, in some research, such as [42, 31, 51], clustering methods have been applied to get several sub-models according to clustered sub-logs. For example, [30] and [10] use data attributes and conformance artifacts for this purpose. Using this approach, the quality of the sub-models is better than a single process model discovered on the complete event log. However, getting several process models may be a barrier for decision-makers who need a single overview of each process.

This research studies the effectiveness of applying biased sampling on event data prior to invoking process discovery algorithms, instead of using all the available event data. In this regard, we present and investigate different biased sampling strategies and analyze their ability to improve process discovery algorithm scalability. Furthermore, the techniques presented allow us to select a user-specified fraction of inclusion of the total available event data. Using the `ProM`-based [48] extension of `RapidMiner` [3], i.e., `RapidProM`, we study the usefulness of these sampling approaches, using real event logs. The experimental results show that applying biased sampling techniques reduces the required discovery time for all discovery algorithms. Moreover, for some event logs, by sampling, we also improve the quality of discovered process models due to implicit filtering.

This paper extends the work [41]. Here, we explain the instance selection strategies in more detail. The proposed approaches are also applied on many real event logs with state-of-the-art process discovery algorithms, i.e., the Alpha Miner, the Inductive Miner, the Split Miner, and the ILP Miner. We propose to select some process instances of an event log based on variants or traces and apply process discovery algorithms on the sampled data. Moreover, in this paper, we used more metrics to evaluate the quality and complexity of discovered process models.

The remainder of this paper is structured as follows. In Section2, we discuss related work. Section 3 defines preliminary notation. We present different biased instance selection strategies in Section 4. The experimental evaluation and corresponding results are given in Section 5. Finally, Section 6 concludes the paper and presents some directions for future work.

## 2. Related Work

Many discovery algorithms, e.g., the Alpha Miner [5], the ILP Miner [53], and the basic Inductive Miner [28] were designed to first build an internal data structure, e.g., the directly follows graph, based on the whole available event data. Subsequently, these algorithms discover a process model directly from this data structure. Other process discovery algorithms, e.g., the Split Miner [7], the Flexible Heuristic Miner [52], the Fuzzy Miner [25], the extended versions of the Inductive Miner[29], and the ILP Miner [54] were designed to be able to filter infrequent or noisy behavior within their internal data structure, prior to discovering a process model. The performance of all these algorithms depends on the number of process instances and the unique number of activities. For some of them, e.g., ILP Miner[53], the number of unique process instances also influences the required time to discover a process model.

Recently, preprocessing of event data has gained attention. In [44, 6], techniques are proposed to increase the quality of discovered process models by cleansing the event data. Some other aim to increase/keep the privacy of users by preprocessing an event log beforehand [40]. Also, in [18], [19], [20] and [21] we have shown that by removing/modifying outlier behavior in event logs, process discovery algorithms are able to discover process models with higher quality. Furthermore, [55] proposes a filtering approach to detect and remove infrequent behavior for event streams. Moreover, [33] uses data attributes to filter out noisy behavior. In [17] an interactive filtering toolkit is provided that let user chooses different filtering methods in combination with several process discovery algorithms. Filtering techniques effectively reduce the size of the event data used by process discovery algorithms. But, to do so, these filtering techniques have non-linear time complexity that does not scale in the context of big data. Meanwhile, sometimes the required time for applying these filtering algorithms is longer than the process discovery time. Also, these techniques have no accurate control over the size of the sampled event log.

Filtering techniques focus on removing infrequent behavior from event data; however, sampling methods aim to reduce the number of process instances and increase the performance of other procedures. Some sampling approaches have been proposed in the process mining field. [26] studies some behavior qualities for sampled event logs. In [22], the authors show that by selecting a few unique process instances of event data, it is possible to approximate the conformance value in a shorter time. In [12], the authors proposed a sampling approach based on the Parikh vector of traces to detect the behavior in the event log. However, we are not able to use this sampling technique for the process discovery purpose, because, the Parikh vector does not store the sequences of activities that are critical for discovering process models. In [8], the authors recommend a random trace-based sam-

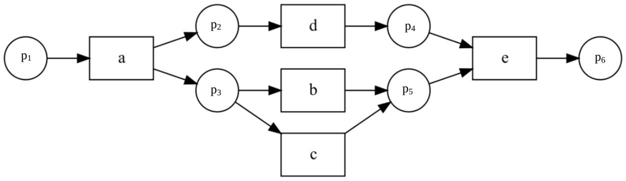


**Fig. 1.** An example process model with Petri net notation.

pling method to decrease the discovery time and memory footprint. This method assumes that process instances have different behavior if they have different sets of directly follows relations. However, using a unique set of directly follows relations may lead to different types of process behavior. Furthermore, [9] recommends a trace-based sampling method specifically for the Heuristic miner[52]. Both these sampling methods are unable to control the size of the final sampled event data. Also, they depend on the defined behavioral abstraction that may lead to the selection of almost all the process instances. Moreover, all these mentioned sampling methods use random trace-based sampling with a replacement that may lead to pick and analyze a unique process instance several times. Finally, as these methods are unbiased, we have non-deterministic results after each sampling. In this paper, we will offer and analyze random and biased instance selection methods which the size of the sampled event data is adjustable.

## 3. Preliminaries

In this section, we briefly introduce basic process mining terminology and notations that ease the readability of the paper. There are different notations to depict a process model. In this paper, we used the Petri net [35] notation. Petri net is a directed bipartite graph that can be defined as follows.

**Definition 1 (Petri net).** *A Petri net is a graph that could be presented as triple (T,P,A) that T is a finite set of transitions, P is a finite set of places and A is a set of arcs that connect places to transitions and transitions to places.* $A \subseteq (T \times P) \cup (P \times T)$.

An example process model with Petri net notation is presented in Figure 1. In this figure, $T = \{a, b, c, d, e\}$, $P = \{p_1, \ldots, p_6\}$, and $A = \{(p_1, a), (a, p_2), \ldots, (e, p_6)\}$.

Given a set $X$, a multiset $M$ over $X$ is a function $M\colon X \rightarrow \mathbf{N}_{\geq 0}$, i.e. it allows certain elements of $X$ to appear multiple times. We write a multiset as $M = [e_1^{k_1}, e_2^{k_2}, ..., e_n^{k_n}]$, where for $1 \leq i \leq n$ we have $M(e_i) = k_i$ with $k_i \in \mathbf{N}_{>0}$. If $k_i = 1$, we omit its superscript, and if for some $e \in X$ we have $M(e) = 0$, we omit it from the multiset notation. Furthermore, $M = [\,]$ denotes an empty multiset. $\overline{M} = \{e \in X \mid M(e) > 0\}$ is the set of elements present in the multiset. The set of all possible multisets over a set $X$ is written as $\mathcal{M}(X)$.

Let $X^*$ denote the set of all possible sequences over a set $X$. A finite sequence $\sigma$ of length $n$ over $X$ is a function $\sigma\colon \{1, 2, ..., n\} \rightarrow X$, alternatively written as $\sigma = \langle x_1, x_2, ..., x_n \rangle$ where $x_i = \sigma(i)$ for $1 \leq i \leq n$. The empty sequence is written as $\epsilon$. The

**Table 1.** Fragment of a fictional event log (each line corresponds to an event).

| Case-id | Activity | Resource | Time-stamp |
|---|---|---|---|
| ... | ... | ... | ... |
| *1* | *register request (a)* | *Karl* | *2017-04-08:08.10* |
| *1* | *examine thoroughly (b)* | *Ali* | *2017-04-08:09.17* |
| *2* | *register request (a)* | *Karl* | *2017-04-08:10.14* |
| *1* | *check resources (c)* | *William* | *2017-04-08:10.23* |
| *1* | *check ticket (d)* | *William* | *2017-04-08:10.53* |
| *2* | *check resources (b)* | *Ali* | *2017-04-08:11.13* |
| *1* | *Send to manager(e)* | *Majid* | *2017-04-08:13.09* |
| *1* | *accept request (f)* | *Fatima* | *2017-04-08:16.05* |
| *1* | *mail decision(h)* | *Anna* | *2017-04-08:16.18* |
| ... | ... | ... | ... |

concatenation of sequences $\sigma$ and $\sigma'$ is written as $\sigma \cdot \sigma'$. Function $hd\colon X^* \times \mathbf{N}_{\geq 0} \nrightarrow X^*$, returns the "head" of a sequence, i.e., given a sequence $\sigma \in X^*$ and $k \leq |\sigma|$, $hd(\sigma, k) = \langle x_1, x_2, .., x_k \rangle$, i.e., the sequence of the first $k$ elements of $\sigma$. In case $k = 0$ we have $hd(\sigma, 0) = \epsilon$. Symmetrically, $tl\colon X^* \times \mathbf{N}_{\geq 0} \nrightarrow X^*$ returns the "tail" of a sequence and is defined as $tl(\sigma, k) = \langle x_{n-k+1}, x_{n-k+2}, ..., x_n \rangle$, i.e., the sequence of the last $k$ elements of $\sigma$, with, again, $tl(\sigma, 0) = \epsilon$. Sequence $\sigma'$ is a subsequence of sequence $\sigma$, which we denote as $\sigma' \in \sigma$, if and only if $\sigma_1, \sigma_2 \in X^*$ such that $\sigma = \sigma_1 \cdot \sigma' \cdot \sigma_2$. Let $\sigma, \sigma' \in X^*$. We define the frequency of occurrence of $\sigma'$ in $\sigma$ by $freq\colon X^* \times X^* \to \mathbf{N}_{\geq 0}$ where $freq(\sigma', \sigma) = |\{1 \leq i \leq |\sigma| \mid \sigma'_1 = \sigma_i, ..., \sigma'_{|\sigma'|} = \sigma_{i+|\sigma'|}\}|$. For example, $freq(\langle b \rangle, \langle a, b, b, c, d, e, f, h \rangle) = 2$ and $freq(\langle b, d \rangle, \langle a, b, d, c, e, g \rangle) = 1$, etc.

Event logs describe sequences of executed business process activities, typically in the context of some cases (or process instances), e.g., a customer or an order-id. The execution of an activity in the context of a case is referred to as an *event*. A sequence of events for a specific case is also referred to as a *trace*. Thus, it is possible that multiple traces describe the same sequence of activities, yet, since events are unique, each trace itself contains different events. An example event log is presented in Table 1.

Consider the events related to *Case-id* value 1. Karl *registers a request*, after which Ali *examines it thoroughly*. William *checks the ticket* and *checks resources*. Ava *sends the request to the manager* and Fatima *accepts the request*. Finally, Anna *emails the decision* to the client. The example trace is written as $\langle a, b, c, d, e, f, h \rangle$ (using short-hand activity names). In the context of this paper, we formally define event logs as a multiset of sequences of activities.

**Definition 2 (Event Log).** *Let $\mathcal{A}$ be a set of activities. An event log is a multiset of sequences over $\mathcal{A}$, i.e. $L \in M(\mathcal{A}^*)$.*

For example, $L_1 = [\langle a, b, c, e, g \rangle^6, \langle a, c, b, e, g \rangle^4, \langle a, b, c, e, f \rangle^3, \langle a, c, b, e, f \rangle^2, \langle a, d, e, f \rangle, \langle a, d, e, g \rangle, \langle a, b \rangle, \langle b, d, c, f \rangle \langle a, b, c, e, e, f \rangle]$ is an event log with 20 traces. Observe that each $\sigma \in \overline{L}$ describes a *trace-variant* whereas $L(\sigma)$ describes how many traces of the form $\sigma$ are presented within the event log. Therefore, in the above event log, there are 9 trace-variants and $L_1(\langle a, c, b, e, g \rangle) = 4$.

Sampling could be done with/without replacement. In sampling with replacement, it is possible to select the sampled objects more than one time. However, in this work, we use sampling without replacement. In other words, it is not possible to put a process instance more than once in the sampled event log. In the following, we define formally sampled event logs.

**Definition 3 (Sampled Event Log).** *We define $S_L$ as a trace-based sampled event log of an event log L, if for each $\sigma \in S_L$, $S_L(\sigma) \leq L(\sigma)$. In the same way, $S_L$ is a variant-based sampled event log of L if for all $\sigma \in S_L$, $S_L(\sigma) = 1$.*

In a variant-based sampled event log, the frequency of each selected process instances is equal to 1, however, in a trace-based event log, it is possible to have more than one traces of a unique process instance. In other words, a variant-based sampled event log is a subset of trace-variants in $L$. For example, $L_2 = [\langle a, b, c, e, g \rangle, \langle a, c, b, e, g \rangle, \langle b, c, e, f \rangle]$ is a variant-based sampled event log of $L_1$.

We could define different types of behavior in an event log. One possible behavior in an event log is the directly follows relation between activities that can be defined as follows.

**Definition 4 (Directly Follows Relation).** *Let $a$ and $b \in \mathcal{A}$ be two activities and $\sigma = \langle \sigma_1, .., \sigma_n \rangle$ is a trace in the event log. A directly follows relation from a to b exists in trace $\sigma$, if there is $i \in \{1, .., n-1\}$ such that $\sigma_i = a$ and $\sigma_{i+1} = b$ and we denote it by $a >_\sigma b$.*

For example, in $\sigma = \langle a, b, c, e, g \rangle$, we have $c >_\sigma e$, but $d \not>_\sigma a$.

An alternative behavior which has negative affects the results of process discovery algorithms is the occurrence of a low probable sub-pattern, i.e., a sequence of activities, between pairs of frequent surrounding behavior, which we refer to it as behavioral contexts [20].

**Definition 5 (Behavioral Context).** *A behavioral context c is a pair of sequences of activities, i.e., $c \in \mathcal{A}^* \times \mathcal{A}^*$. Furthermore, we define the set of behavioral contexts present in L, i.e., $\beta_L \in \mathcal{P}(\mathcal{A}^* \times \mathcal{A}^*)$, as:*

$$\beta_L = \{(\sigma_l, \sigma_r) \in \mathcal{A}^* \times \mathcal{A}^* \mid \exists \sigma \in L, \sigma' \in \mathcal{A}^* \, (\sigma_l \cdot \sigma' \cdot \sigma_r \in \sigma)\} \tag{1}$$

For example, in trace $\sigma = \langle a, b, c, e, g \rangle$, $\langle a, b \rangle$ and $\langle e \rangle$ are two subsequences that surround $\langle c \rangle$, hence, the pair $(\langle a, b \rangle, \langle e \rangle)$ is a behavioral context.

We inspect the probability of contextual sub-patterns, i.e., the behavior that is surrounded by frequent behavioral contexts. For this purpose, we simply compute the empirical conditional probability of a behavioral sequence, being surrounded by a certain context.

**Definition 6 (Conditional Contextual Probability).** *Let $\sigma_s, \sigma_l, \sigma_r \in \mathcal{A}^*$ be three sequences of activities and let $L \in \mathcal{M}(\mathcal{A}^*)$ be an event log. We define the* conditional contextual probability *of $\sigma_s$, w.r.t., $\sigma_l$ and $\sigma_r$ in L, i.e., representing the sample based estimate of the conditional probability of $\sigma_s$ being surrounded by $\sigma_l$ and $\sigma_r$ in L. Function $\gamma_L : \mathcal{A}^* \times \mathcal{A}^* \times \mathcal{A}^* \to [0, 1]$, is based on:*

$$\gamma_L(\sigma_s, \sigma_l, \sigma_r) = \frac{\sum\limits_{\sigma \in L} \left( |\sigma_{\sigma_l \cdot \sigma_s \cdot \sigma_r}| \right)}{\sum\limits_{\sigma \in L} \Big( \sum\limits_{\sigma' \in \mathcal{A}^*} |\sigma'_{\sigma_l \cdot \sigma' \cdot \sigma_r}| \Big)} \tag{2}$$

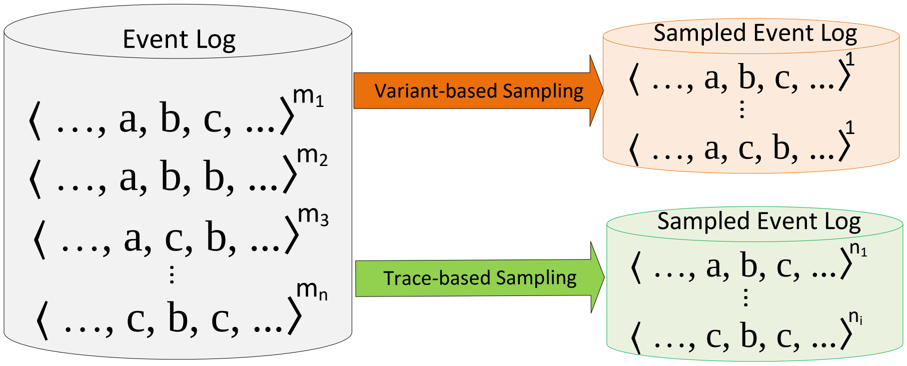

Event Log
⟨ …, a, b, c, … ⟩$^{m_1}$
⟨ …, a, b, b, … ⟩$^{m_2}$
⟨ …, a, c, b, … ⟩$^{m_3}$
⟨ …, c, b, c, … ⟩$^{m_n}$
Variant-based Sampling
Trace-based Sampling
Sampled Event Log
⟨ …, a, b, c, … ⟩$^{1}$
⟨ …, a, c, b, … ⟩$^{1}$
Sampled Event Log
⟨ …, a, b, c, … ⟩$^{n_1}$
⟨ …, c, b, c, … ⟩$^{n_i}$

**Fig. 2.** Schematic overview of instance selection (sampling) methods. We select variants or traces based on different criteria. In variant-based sampling, each variant in a sampled event log represents all the traces with this behavior in the original event log.

*We alternatively write* $P_L(\sigma_s \mid \sigma_l, \sigma_r)$ *to represent* $\gamma_L(\sigma_s, \sigma_l, \sigma_r)$.

For example, to compute $P_{L1}(\langle c\rangle \mid \langle a, b\rangle, \langle e\rangle)$, we need to compute in how many traces, subsequence $\langle a, b, c, e\rangle$ is occurred and divide it to the number of traces that contain the pair $(\langle a, b\rangle, \langle e\rangle)$ as a behavioral context that is $\frac{10}{10} = 1$. Based on these probabilities, we are able to detect unstructured behavior in a trace.

## 4. Sampling Event Data using Instance Selection

In this section, we present different biased instance selection strategies to sampling event logs and consequently increase the discovery procedure's performance. We are able to sample different behavioral elements of an event log, e.g., events, directly follow relations, traces, and variants. By sampling events, it is possible to choose events from different parts of a process instance that is harmful to the process discovery purpose. Sampling directly follows relations is useful for some process discovery algorithms like the Alpha Miner. For example, choosing the most frequent directly follows relations. But, we need to modify these algorithms to accept a set of directly follows relations instead of an event log as an input. However, it may result in process models with some unconnected components. Also, such data structures are not applicable to all process discovery algorithms. Thus, here we just consider trace and variant-based sampling. Therefore, here we just focus on sampling methods that take an event log as an input and return a subset of traces or variants. In other words, the sampling methods select some variants/traces in the input event logs.

The schematic of the instance selection methods is illustrated in Figure 2. In variant-based sampling, the frequency of each sample is 1. In other words, we select only one process instance for each selected variant. However, in trace-based sampling, the frequency of each unique sample is $1 \leq n_i \leq m_i$.

For many process discovery algorithms such as the ILP Miner, the family of Alpha miners and the Inductive Miner, it is enough to have unique variants to discover a corresponding process model. The frequency of variants is mostly just used for post-processing algorithms like filtering. Therefore, here we mainly focus on variant-based sampling, but, all these methods can easily be extended to trace-based sampling methods. We also just used control-flow related information that is available in all event logs, and this is consistent with the way.

We are able to consider three dimensions for a sampled event log. The first one is the number of process instances that are placed in the sampled event log, i.e., $|S_L|$. In the worst case, it is the same as the original event log, i.e., no reduction in size. We can set the size of sampled event logs (i.e., the sample ratio) as follows.

$$c = \begin{cases} \frac{|S_L|}{|\overline{L}|} & \textit{Variant-based sampling} \\ \frac{|S_L|}{|L|} & \textit{Trace-based sampling} \end{cases} \tag{3}$$

Note that, in the above equation, $0 < c \leq 1$ and denotes that how many percentages of traces (or variants in variant-based sampling) are selected in the sampled event log.

Another dimension is the completeness of the sampled event log. If a sample event log contains few relations of the original event log, process discovery algorithms are not able to discover an appropriate process model from the sampled event log. However, it is not required that sampled event logs include all the behavior in the original event log, because many process discovery algorithms are able to generalize the seen behavior. Finally, the last dimension is the sampling time as a preprocessing phase. It is better to sample an event log in a shorter time.

It is possible to sample process instances or variants of an event log randomly or selecting them based on a bias. In the following, we will explain both of these methods.

### 4.1. Random Sampling

The first method is to randomly sample $c \times |L|$ traces in the event log without replacement and return these traces (i.e., trace sampling) or just unique trace-variants among them (i.e., variant sampling). The time complexity of this method is $O(k)$ where $k$ is the number of sampled traces. Therefore, this method is fast because there is no need to traverse the original event log. In this way, we have no control over the number of selected variants. In other words, it is possible that many of the sampled traces have similar behavior and we return just a few unique variants.

Another approach is to first find all the unique variants in an event log, after that, randomly select $c \times |\overline{L}|$ variants from them. The time complexity of traversing an event log is $O(n \times l)$ where $n = |L|$ is the number of traces, and $l$ is the average length of traces in the event log. This approach is a bit slower, but it is able to return more behaviors compared to the previous approach.

### 4.2. Instance Selection Strategies

In general, traversing an event log is not a very time-consuming phase, and this gives us a motivation that instead of randomly sampling the variants, we are able to use more advanced strategies (biases) to select variants in the event log. Note that although in XES standard [48], we need to traverse an event log to find the variants, in many other structures variants with some of their properties are stored as meta-data [16].

In biased sampling methods, we first traverse the event log to find unique trace-variants in it, then rank them based on different strategies. Afterward, we return the top $c \times |\overline{L}|$ variants with the highest rank in the sampled event log. For the trace-based sampling methods, we return all process instances that correspond to each trace-variant (with considering the $c$ value). We are able to use different ranking strategies for selecting the trace-variants that will be discussed in follows.

### Frequency-based Sampling

The first ranking strategy is sampling variants based on their frequencies. This sampling method gives more priority to a trace-variant that has a higher occurrence frequency in the event log. Thus, we sort the variants based on their frequencies or $L(\sigma)$ and return the top $c \times |\overline{L}|$ of variants as a sampled event log. For example, for $L_1$ if we want to select only one variant based on their frequencies, $\langle a, b, c, e, g \rangle$ will be selected. The time complexity of this strategy is $O(n{\times}l + r)$ where $n{\times}l$ corresponds to traversing the event log and $r$ represents the required time that for ranking variants $\sigma{\in}\bar{L}$.

The advantage of this strategy is that we are able to have a minimum replay fitness of the future process model that will be discovered based on the sampled event log. Note that, in the random sampling strategy, the probability of choosing a more frequent variant is also higher. However, in some event logs, there are lots of variants with very low frequencies. Sometimes, the majority of process instances have a unique trace-variant. Thus, differentiating between them will be challenging. Therefore, a drawback of this strategy is that the sampled event log may not contain many behaviors in the original event log. The trace-based version of this instance selection bias is existed in many process mining tools. They used this selection bias to reduce the complexity of the resulted process model and show the main stream behavior of the process model.

### Length-based Sampling

We are able to select process instances or trace-variants based on their length, i.e., the number of activities in the trace or $|\sigma|$. If we want to keep more behaviors in our sampled event log, we need to select longer traces first. However, if we are interested in retaining the main-stream behaviors of the event log, it is usually better to choose shorter variants. For the running example, we will chose $\langle a, b, c, e, e, f \rangle$ and $\langle a, b \rangle$ first if we use longer and shorter strategies respectively.

Selection based on the *longer* strategy, we are able to leave out incomplete traces, that improves the quality of resulted process models. However, if there are self-loops and other loops in the event log, there is a high probability to consider many infrequent variants with the same behavior for process discovery algorithms. For example, if we use directly follows information, it does not matter if trace $\sigma$ has $a >_\sigma a$ one time or more. On the other hand, with selection based on *shorter* strategy, if there are incomplete traces in an event log, we probably will have them in the sampled event log. Similar to the frequency-based strategy, the time complexity of this approach is $O(n{\times}l + r)$.

**Similarity-based Selection**

If we aim to sample variants that contain general behavior of the whole event log, we need to use the similarity-based sampling methods. In this approach, we first find the general behaviors of the event log. We are able to use different behavior, however, the simplest and the most critical behavior for process discovery is the directly follows relation. Therefore, we compute the occurrence probability of each directly follows relation $B_i = (a, b)$ (that $a, b \in \mathcal{A}$) according to the following formula.

$$Prob(B_i) = \frac{|\{\sigma \in \overline{L} : a >_\sigma b\}|}{|\overline{L}|} \tag{4}$$

Therefore, we compute the probability of observing each directly follows relation $B_i$ in a variant. If $Prob(B_i)$ is high (i.e., be higher than a defined threshold $0 \geq T_P \geq 1$), we expect that sampled variants also should contain it. Thus, any variant contains such a high probable behavior (that here is a directly follows relations), will give a $+1$ to its rank. Otherwise, if a variant does not contain a probable behavior, we decrease its rank by $-1$. For example, $(e, f)$ is a high probable relation with $Prob(e, f) = \frac{5}{9}$. Note that by using a higher $T_P$ value, we will have fewer high probable and low probable relations; therefore, many variants will have similar ranks.

Contrariwise, if a variant contains a low probable behavior (i.e., $Prob_{B_i} \leq 1 - T_P$), we decrease its rank by $-1$. In other words, we are searching for variants that have much high probable behaviors and have less low probable behaviors. For example, $Prob(e, e) = \frac{1}{9}$ is a low probable relation. Note that, it is possible that some behaviors be neither high probable nor low probable that we do nothing for such behaviors. Afterward, we sort the variants based on their ranks and return the $c \times |\overline{L}|$ ones with the highest rank. For example, $\langle a, d, e, f \rangle$ is the first variant that will be chosen using this strategy for log $L_1$.

The main advantage of this method is that it helps process discovery algorithms to depict the main-stream behavior of the original event log in the process model. However, it needs more time to compute a similarity score of all variants. Especially, if we use more advanced behavioral structures such as eventually follow relations, this computation will be a limitation for this ranking strategy. Another disadvantage of this strategy is that the length of variants affects the final rank. It is possible that two variants have a similar set of directly follows relations, but based on the frequency of these directly follows, they will have different ranks. The time complexity of this strategy is $O(n \times l + d + r)$ where $d$ corresponds to finding probabilities of directly follows relations and in the worst case, its complexity is $O(|A|^2 \times \bar{L})$.

**Structured-based Sampling**

In this sampling method, we consider the presence of unstructured behavior (i.e., based on Definition 6) in each variant. In this regard, we first compute the occurrence probability of each sub-pattern among its specific contextual context (i.e., $P_L(\sigma_s, \sigma_l, \sigma_r)$). If this probability is below a given threshold, i.e., $0 \geq T_S \geq 1$, we call it an odd structure. We expect that unstructured subsequences are the most problematic behavior in event logs that make discovered process models inaccurate and complex [20]. Thus, for each unstructured behavior in a trace-variant, we give a penalty to it and decrease its rank by $-1$. For example, $P_{L1}(\langle d \rangle | \langle b \rangle, \langle c \rangle)$ is very low and $\langle b, d, c, f \rangle$ will give a penalty for it.

Consequently, a variant with higher odd structures receives more penalties, and it is not appealing to be placed in the sampled event log.

By choosing a high $T_S$ value, most of behaviors will be considered as odd structures. In contrast, a very low $T_S$ will result in few unstructured behaviors that leads to have many variants with the same rank. Note that, by increasing the length of behavioral context, we will detect more important nonstructural behaviors. Similarly, by decreasing the $T_S$ value, the detected behavioral patterns will be more problematic.

Similar to similarity-based selection strategy, in this strategy, the length of variants will affects their rank. Using this strategy, we select the most well-structured trace-variants in the event log, that is expected to have process models with high precision value. However, it is possible to select trace-variants that not frequent in the input event log. The time complexity of this strategy is $O(n{\times}l + c + r)$ where $c$ corresponds to conditional contextual probabilities and in the worst case, its complexity is $O(|A|^m{\times}\bar{L})$ where $m$ is the maximum length of considered contextual behavior and the substring. Therefore, this strategy is potentially the slowest strategy for instance selection.

**Hybrid Methods**

In a hybrid strategy, we are able to combine two or three of other sampling strategies. In this way, we expect to have benefits of different methods. Here, we combine the frequency-based and similarity-based methods. In this regard, when we compute the probability of each directly follows relation, the frequency of it in the whole event log will be used according to the following formula.

$$Prob(a,b) = \frac{\Sigma_{\sigma\in\bar{L}_{a>b}}\ (L(\sigma))}{|L|} \tag{5}$$

In the above equation, $\bar{L}_{a>b} = \{\sigma{\in}\bar{L} :\ a >_{\sigma} b\}$. Note that, other combinations are also possible.

Different ranking strategies require different sampling time and result in different sampled event logs according to their completeness. In the next section, we will analyze how these selection strategies affect the performance and quality of process discovery algorithms on real event logs.

## 5. Evaluation

In this section, by doing experiments, we aim to answer the following research questions:

- **Q1**) Are the proposed instance selection strategies able to improve the performance of process discovery?
- **Q2**) What are the effects of instance selection on the quality of discovered process models?

In this regard, we first explain the implementation details of the proposed method. Afterward, the experimental settings for the experiments will be discussed, and finally, the results will be explained.

**Table 2.** Some information of real event logs that are used in the experiment.

| **Event Log** | **Activities#** | **Traces#** | **Variants#** | **Directly Follows Relations#** |
|---|---|---|---|---|
| *BPIC*_2012 [45] | 23 | 13087 | 4336 | 138 |
| *BPIC*_2013 [49] | 4 | 7554 | 1511 | 11 |
| *BPIC*_2017_*All* [46] | 26 | 31509 | 1593 | 178 |
| *BPIC*_2017_*Offer* [46] | 8 | 42995 | 169 | 14 |
| *BPIC*_2018_*Control* [15] | 7 | 43808 | 59 | 12 |
| *BPIC*_2018_*Inspection* [15] | 15 | 5485 | 3190 | 67 |
| *BPIC*_2018_*Reference* [15] | 6 | 43802 | 515 | 15 |
| *BPIC*_2019 [47] | 44 | 251734 | 11973 | 538 |
| $Hospital$ [32] | 18 | 100000 | 1020 | 143 |
| $Road$ [14] | 11 | 150370 | 231 | 70 |
| $Sepsis$ [33] | 16 | 1050 | 846 | 115 |

## 5.1. Implementation

To apply the proposed instance selection strategies, we implemented the *Sample Variant* plug-in in `ProM` framework[3]. ProM is an academic platform with several plug-ins that cover wide domains of process mining areas [4]. In this implementation, we used static thresholds for both similarity and structure-based ranking strategies. For similarity-based selection, we just considered the directly follows relation as explained in the previous section. Using this plug-in, the end-user is able to specify her desired percentage of sampling variants/traces and the instance selection strategy. It takes an event log as an input and returns the top fraction of its variants/traces. A screen-shot of the settings of the instance selection plug-in in ProM is presented in Figure 3. By adjusting these settings, users will be able to have different sampled event logs.

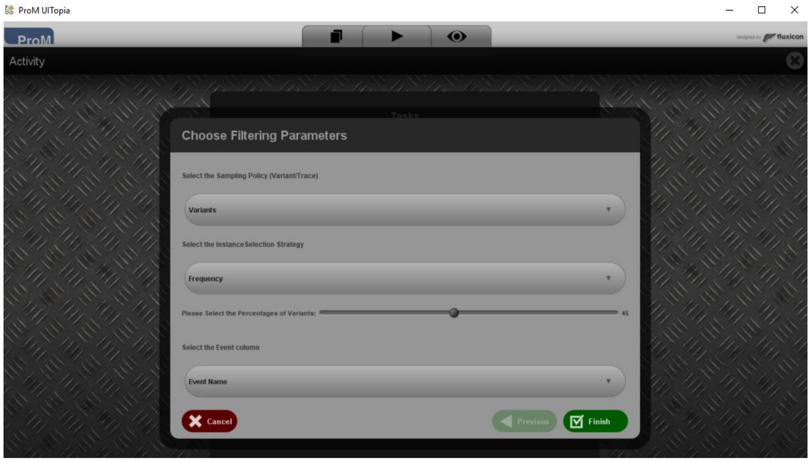
In this implementation, we used $T_P$=0.8 and $T_S$=0.2. Moreover, for behavioral contexts' subsequences, we consider the maximum length equals to 2.

Furthermore, to apply our proposed method on various event logs and use different process discovery algorithms with their different parameters, we ported the *Sample Variant* plug-in to `RapidProM` [3] which extends `RapidMiner` with process analysis capabilities. In our experiments, we also used the statistical sampling method that is presented in [8]; however, as we consider only work-flow information, its relaxation parameter is ignored.

## 5.2. Experimental Setup

For analyzing the effect of sampling, we applied the proposed method on several real event logs. Some information about these event logs is given in Table 2. We added artificial *start* and *end* activities to the whole process instances of all event logs. For process discovery, we used the Alpha Miner [5], the Inductive Miner (IM) [28], the Inductive Miner with its infrequent behavior filtering mechanism (IMi) [29], the ILP Miner [43], and the Split Miner [7]. For computing the quality of discovered process models, the original event logs were used.

[3] Sample Variant plug-in in: svn.win.tue.nl/repos/prom/Packages/LogFiltering.



Choose Filtering Parameters

Select the Sampling Policy (Variant/Trace)

Variants

Select the InstanceSelection Strategy

Frequency

Please Select the Percentages of Variants: 45

Select the Event column

Event Name

**Fig. 3.** A screen-shot of the implemented instance selection plug-in in ProM platform. It receives an event log and return a sample event log based on the adjusted settings.

We sampled event logs with different variant and trace-based instance selection strategies, and $c$ with different values that are $[1, 2, 3, 5, 10, 15, 20, 25, 30, 40, 50, 75, 100]$ which $c$ presents how many percentages of traces or variants of the event log is selected to be placed in the sampled event log. Therefore, if we use the trace-based sampling policy with $c = 100$, the sampled event log equates to the original event log. As sampling time and discovery time are nondeterministic, each experiment was repeated four times.

### 5.3. Experimental Result

Here, we show how experimental results address the mentioned research questions.

**Performance Analysis**

To measure the improvement in the performance, we consider both the discovery time and sampling time of event logs using the following formulas:

$$DiscoveryTimeImprovement = \frac{DiscoveryTime_{WholeLog}}{DiscoveryTime_{SampledLog}} \tag{6}$$

$$TotalTimeImprovement = \frac{DiscoveryTime_{WholeLog}}{DiscoveryTime_{SampledLog} + SamplingTime} \tag{7}$$

In discovery time improvement, we measure how discovering a process model is faster on a sampled event log compared to the original event log. However, in total time improvement, the sampling time is also considered. Therefore, it measures how sampling an event log and discovering a process model of it, is faster compared to discover a process model from the original event log. An improvement value shows how many times a process model will be discovered faster by using the sampled event log.

**Table 3.** Process discovery performance improvement for different process discovery algorithms using variant and trace-based instance selection methods when $c = 10$.

| Discovery Algorithm | Event Log | Normal Discovery Time (ms) | Discovery Improvement | | | Discovery Algorithm | Normal Discovery Time (ms) | Discovery Improvement | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Statistical | Traces | Variants | | | Statistical | Traces | Variants |
| Alpha Miner | BPIC-2012 | 154 | 4 | 7 | 50 | Inductive Miner (IMi) | 3,571 | 4 | 7 | 18 |
| | BPIC-2013 | 94 | 6 | 8 | 17 | | 890 | 9 | 11 | 48 |
| | BPIC-2017-All | 1,349 | 7 | 9 | 38 | | 70,652 | 11 | 27 | 50 |
| | BPIC-2017-Offer | 303 | 38 | 14 | 758 | | 2,108 | 44 | 14 | 727 |
| | BPIC-2018-Control | 234 | 585 | 9 | 41 | | 1,707 | 43 | 9 | 557 |
| | BPIC-2018-Inspection | 252 | 2 | 10 | 45 | | 4,522 | 2 | 16 | 39 |
| | BPIC-2018-Reference | 186 | 24 | 9 | 61 | | 1,511 | 28 | 7 | 184 |
| | BPIC-2019 | 1,817 | 14 | 6 | 25 | | 62,021 | 16 | 10 | 42 |
| | Hospital | 687 | 14 | 8 | 224 | | 17,366 | 35 | 29 | 1,192 |
| | Road | 660 | 42 | 6 | 1,650 | | 6,988 | 68 | 11 | 12,332 |
| | Sepsis | 20 | 1 | 7 | 51 | | 317 | 1 | 51 | 48 |
| ILP Miner | BPIC-2012 | 2,4875 | 4 | 6 | 9 | Split Miner | 903 | 3 | 5 | 8 |
| | BPIC-2013 | 1,108 | 4 | 4 | 7 | | 537 | 6 | 8 | 25 |
| | BPIC-2017-All | 1,088,584 | 30 | 14 | 41 | | 11,479 | 9 | 13 | 31 |
| | BPIC-2017-Offer | 2,279 | 7 | 6 | 12 | | 1,599 | 25 | 9 | 147 |
| | BPIC-2018-Control | 2,511 | 11 | 6 | 19 | | 1,964 | 36 | 8 | 353 |
| | BPIC-2018-Inspection | 7,746 | 1 | 6 | 5 | | 1,755 | 2 | 7 | 17 |
| | BPIC-2018-Reference | 2,390 | 9 | 5 | 14 | | 1735 | 27 | 7 | 294 |
| | BPIC-2019 | 95,697 | 5 | 3 | 7 | | 19,520 | 15 | 10 | 41 |
| | Hospital | 10,995 | 5 | 7 | 8 | | 3,388 | 13 | 8 | 64 |
| | Road | 6,144 | 10 | 5 | 10 | | 3601 | 46 | 8 | 171 |
| | Sepsis | 7,464 | 1 | 5 | 4 | | 180 | 1 | 4 | 3 |

Table 3 and Figure 4 show the improvement when we sample event logs. The y-axis in these figures represents the average performance improvements using a logarithmic scale. Values below 1 show that there is no improvement in performance. Here, we consider $c = 10$ and all instance selection strategies. *Statistical* corresponds to the sampling method that is proposed in [8] (with its default setting). It is clear that by reducing the size of an event log, the required process discovery time is reduced. Therefore, the $DiscoveryTimeImprovement$ for variant-based selection strategies is significantly higher than the trace-based policy. Note that all process discovery algorithms traverse the event log at least once that requires $O(n{\times}l)$. Therefore, $n = |S_L|$ for variant-based sampled event log is usually extremely lower than the trace-based one. For some event logs, a process discovery algorithm is more than 1000 times faster on the sampled event log using variant-based sampling compared to using the whole event logs. However, for event logs such as *Sepsis*, where most of the traces have unique control-follow related behavior in the original event log, trace-based sampling methods are faster. We see that the Statistical method is not able to improve the total discovery time of *Sepsis* event log for any process discovery algorithm. As it is shown in Table 2, almost all of trace-variants in this event log are unique that cause this sampling method selects almost all the traces in the event log. For the Alpha miner, we usually do not have improvement. This process discovery algorithm is simple, and its main time-consuming task is to find out the directly follows relations in the event log. However, for advanced process discovery algorithms, e.g., the Inductive Miner, the total improvement is very high. Comparing these two figures, we find that the sampling time for the statistical sampling method is much faster than our techniques, but the discovery time of sampled event logs is less using the proposed instance selection strategies.

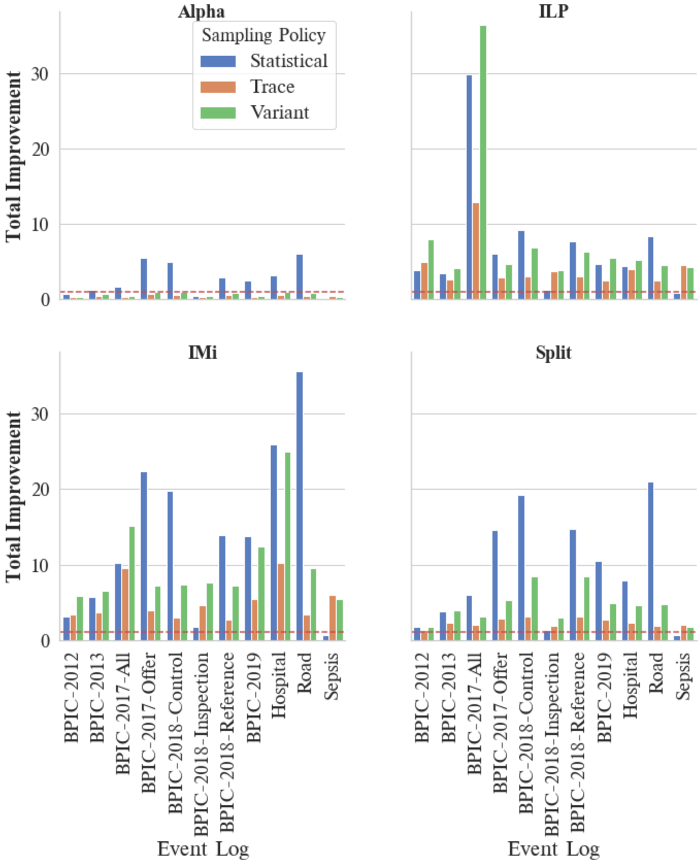


**Fig. 4.** Total time improvement for discovering process using sampling methods.

Figure 5 shows improvements in the performance of process discovery algorithms when we select some instances of event logs. Here, for instance selection strategies, we used the sampling threshold (i.e., $c$) equals to $0.1$ and the variant-based policy. It shows that the improvement is usually lower when the statistical sampling method even it does not need to traverse the input event log. As it is mentioned before, for *Sepsis* event log, this method has no improvement. Among instance selection methods, *Longer* strategy usually results in lower improvement. The computation time of process discovery algorithms depends on different parameters e.g., the number of activities (i.e., $|A|$), the number of variants (i.e., $|\bar{L}|$) and sometimes the prefix-closure of the given event log. Using instance selection strategies, we are able to reduce all the above parameters and consequently decrease the required time for process discovery.

The sampling time of different methods is depicted in Figure 6[4]. As we expected, except for the *Sepsis* event log, the statistical sampling method is much faster than instance

[4] We do not show the results for all event logs.

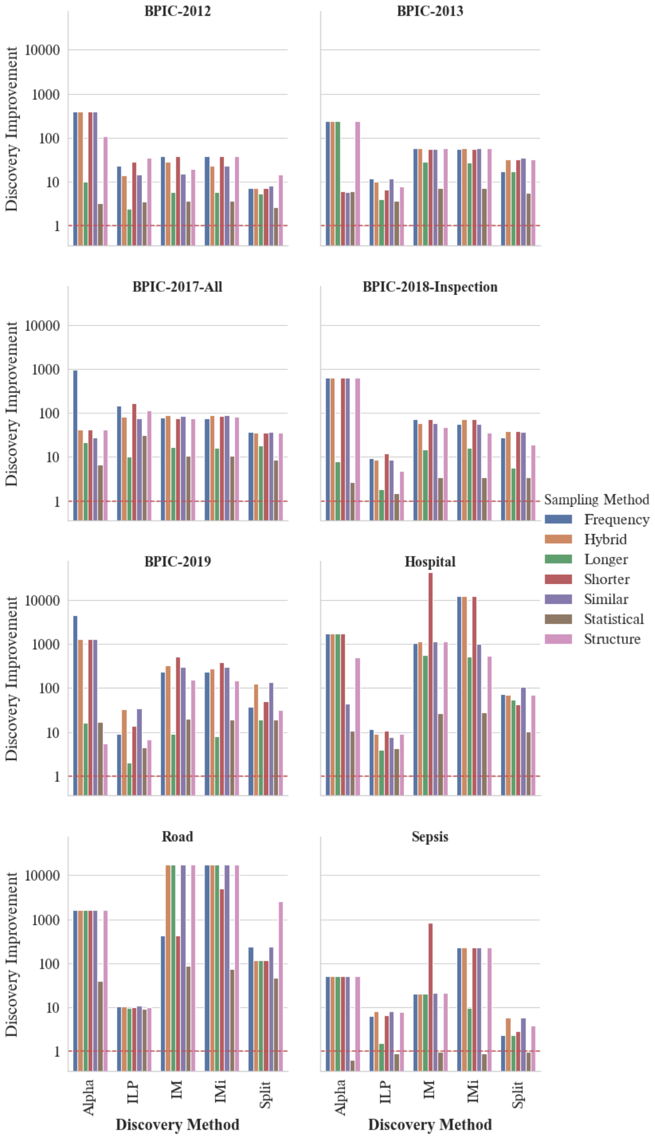


**Fig. 5.** Discovery time improvement for discovering process using instance selection.

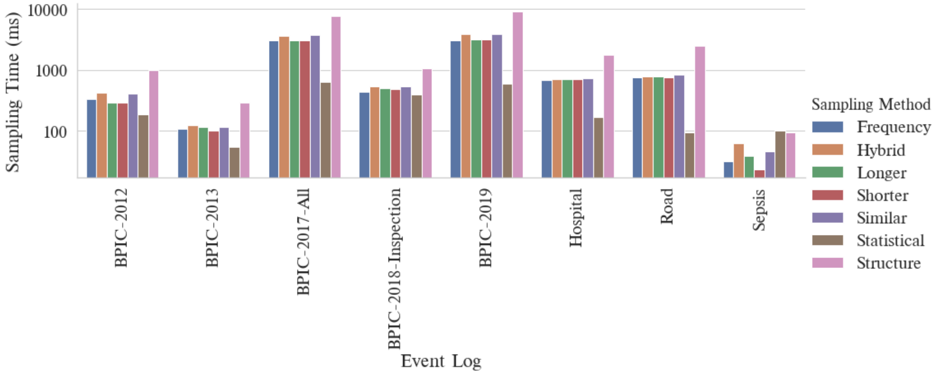


**Fig. 6.** Sampling time of different instance selection strategies.

selection strategies as it does not need to traverse the event log. Among the instance selection strategies, the structure-based is the slowest one as it needs to consider more complex behavioral structures. Frequency and length based strategies are required almost the same time to sampled event logs. As it is mentioned, in case that we use MXML or any other structure that contains meta-data of variants, the sampling time for these strategies will be much less.

As event filtering algorithms are able to reduce the variability of event logs, we are able to compare them with the instance selection methods. In this regard, we filter event logs using Anomaly Free Automaton (AFA) [13] and Matrix Filter (MF) [18] using their default settings. For sampling, we used the variant-based sampling using $c = 10$ and the similarity based strategy. Here, for process discovery, we used the IMi algorithm with its default setting. The results of this experiment is shown in Table 4. We could not have the filtered event log for *BPIC-2018-Inspection* using the AFA method. Results show that the preprocessing time for the instance selection method is less than other filtering methods. For most of the event logs, process discovery on the preprocessed event log using the proposed approach is faster mostly because it returns the variant-based sampled event logs. The discovery time for some filtered event logs is significantly low that is because of removing lots of behavior during filtering.

We aim to analyze what is the origin of the performance improvement. The improvement in performance of process discovery algorithms may be driven by reducing (1) the number of activities, (2) the number of traces, or (3) the amount of unique behavior (e.g., DF relations or variants). By event log instance selection, it is possible that some of the infrequent activities are not placed in the sampled event log. Moreover, by sampling we reduce the size of event logs (i.e., $|S_L| \leq |L|$), specifically when we apply the variant-based strategies. Finally, it is also possible that we reduce the number of unique behavior in the event log.

Figure 7 shows the process discovery time of sampled event logs with different instance selection thresholds when we apply the ILP miner. Here, to focus on the reason for improvements, we used variant-based strategies. When we used the sampling size equals to 100, we will have all the behaviors of the original event log in the sampled event log;

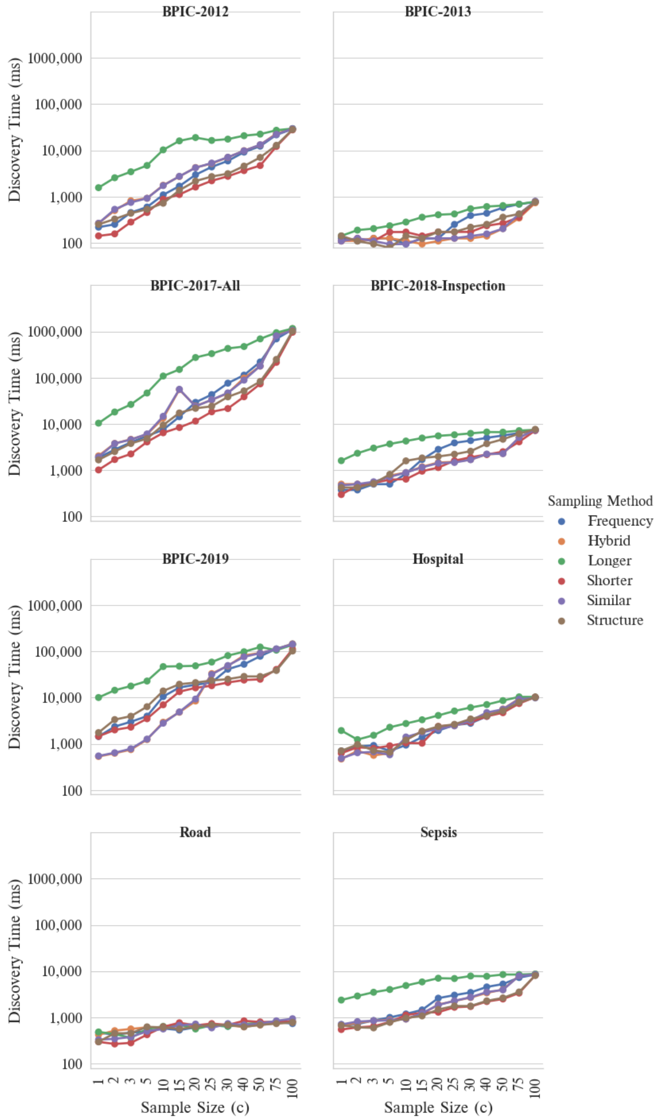


**Fig. 7.** The average of discovery time of the ILP miner for different instance selection methods with different sampling size (i.e., *c*).

**Table 4.** Comparing the instance selection and filtering methods based on the required time of them for preprocessing and process discovery.

| | Preprocessing Time | | | Discovery time | | | Total Discovery Time | | |
|---|---|---|---|---|---|---|---|---|---|
| Event Log | AFA | MF | Sampling | AFA | MF | Sampling | AFA | MF | Sampling |
| BPIC-2012 | 26,705 | 593 | 749 | 379 | 19978 | 140 | 27084 | 20571 | 889 |
| BPIC-2013 | 3246 | 312 | 187 | 435 | 749 | 16 | 3681 | 1061 | 203 |
| BPIC-2017-All | 57927 | 3188 | 5212 | 2 | 1034 | 859 | 57929 | 4222 | 6071 |
| BPIC-2018-Inspection | $\sim$ | 1192 | 649 | $\sim$ | 4 | 170 | $\sim$ | 1196 | 819 |
| BPIC-2019 | 804913 | 6663 | 4403 | 7115 | 312 | 219 | 812028 | 6975 | 4622 |
| Hospital | 27442 | 2122 | 828 | 2014 | 19767 | 15 | 29456 | 21889 | 843 |
| Road | 13229 | 2763 | 937 | 3418 | 4478 | 1 | 16647 | 7241 | 938 |
| Sepsis | 12028 | 62 | 47 | 2 | 4384 | 1 | 12030 | 4446 | 48 |

however, the number of traces in the sampled event log is significantly lower than the original event log. The greatest improvement is usually gained when we use the similarity or shorter strategies for this process discovery algorithm. Results show that for many event logs, by increasing the percentage of selected variants, how will reduce the improvement. Therefore, for many event logs, the main reason for the improvement in performance of the process discovery is gained by reducing the number of variants. However, in *Road* and *Hospital* event logs that there are high-frequent variants, reducing the number of traces has a higher impact on the performance of the process discovery algorithms.

As it is explained, the amount of behavior in the event log also has an important role in the performance of process discovery algorithms. The remaining percentage of *DF* relations in the sampled event logs for different instance selection strategies are given in Figure 8 when we increased the size of the selected percentage of variants (i.e., $c$). To compute the remained DF relations, we divide the number of existing relations in the sampled event log to all DF relations in the original event log. We see that for most of the event logs, the similar and structure-based selection strategies keep fewer DF relations. However, according to their ranking methods, they keep the most common DF relations among the original event log. Moreover, it is shown that for many event logs, with selecting only $50\%$ of the variants, we are able to keep more than $80\%$ of the DF relations in the sampled event log when the frequency or shorter strategies are used. Unlike our expectations, by using the shorter strategy for selecting variants, we will have high percentages of Df relations in the sampled event log. It happens because using this strategy, lots of infrequent trace-variants that may be incomplete, are also selected to be kept in sample event logs.

Note that there is no such control like the $c$ value for the statistical sampling method and it aims to keep as much as DF relations in sampled event logs. However, for most of process discovery algorithms variants are more important compared to only DF relations. Even the basic Inductive miner that uses DF relations may result in different process models for event logs that have identical sets of DF relations. For example, $L_2 = [\langle a, b, c \rangle, \langle a, c, b \rangle]$ and $L_3 = [\langle a, c, b, c \rangle, \langle a, b \rangle]$ have the same sets of DF relations; however, their process models are different. Figure 9 indicates that the average percentage of the remaining variants in sampled event logs using the statistical sampling method [8].

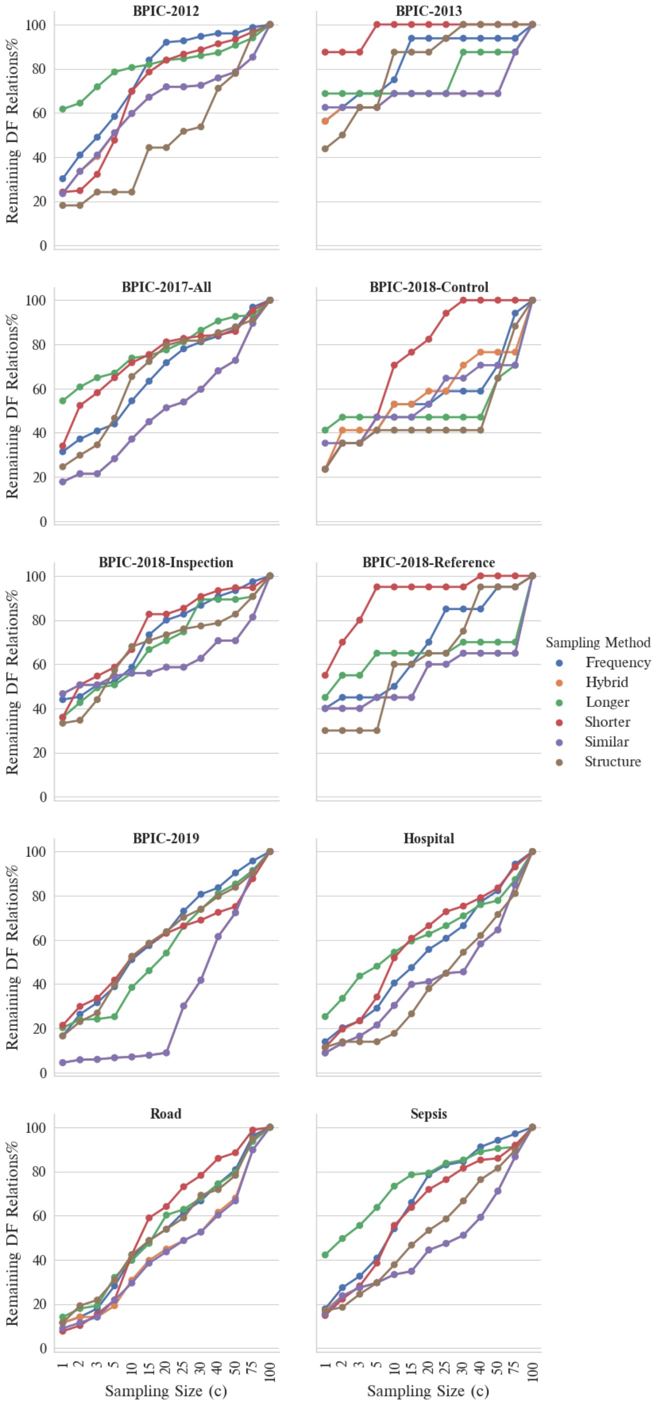


**Fig. 8.** Remained percentage of DF relations in the sampled event logs using instance selection strategies with different sampling size.

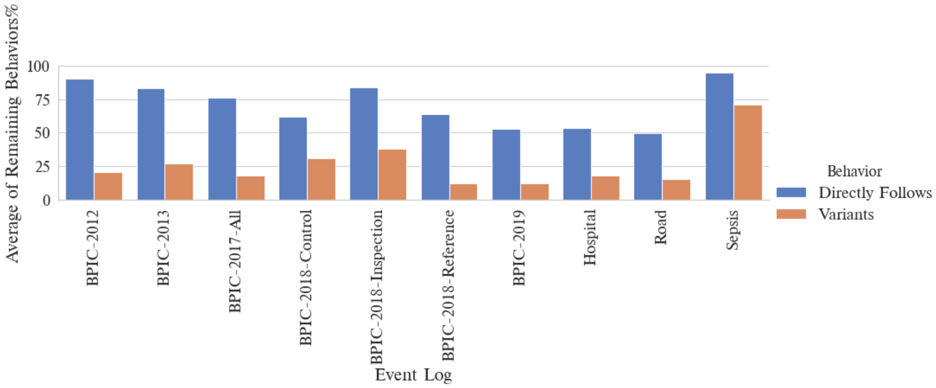


**Fig. 9.** Average of remained variants and directly follows relations in sampled event logs using the statistical sampling method [8].

It shows this method is able to keep a few percentages of variants of an event log (using the default setting).

According to these experiments, we find that using instance selection improves the performance of process discovery algorithms, specifically if we apply the variant-based policy. Moreover, we show that instance selection strategies that only uses the variant meta-data, e.g., frequency or length-based methods are faster than other ones. The instance selection strategies improve the performance of process discovery algorithms by reducing the number of process instances, variants and behaviors in sampled event logs. Note that, many times we consider sampling as a preprocessing phase, and we apply different process mining algorithms, e.g., process discovery several times on the preprocessed event log. As explained in Section 2, there are some preprocessing algorithms that by removing traces with outlier behaviors will reduce the size of the event log. To compare the performance of instance selection strategies with other related preprocessing algorithms, in Figure 10, we examined the average of preprocessing time of three trace filtering methods and instance selection methods. For some event log, we are not able to have the preprocessed event logs. For this experiment, we filter event logs with four different filtering settings and iterate the experiments for four times. The result shows that the instance selection method preprocessed the event logs much faster. Note that, in these filtering methods, we do not have accurate control over the size of the preprocessed event logs.

## Quality Analysis

There are some research has been done on measuring the quality of business process models [34, 23, 37]. Here, to analyze the quality of process models that are discovered on sampled event logs, we use *fitness* and *precision* metrics. Fitness measures how much behavior in the event log is also described by the process model. Thus, a fitness value equal to 1, indicates that all behavior of the event log is described by the process model. Precision measures how much of behavior, that is described by the process model, is also

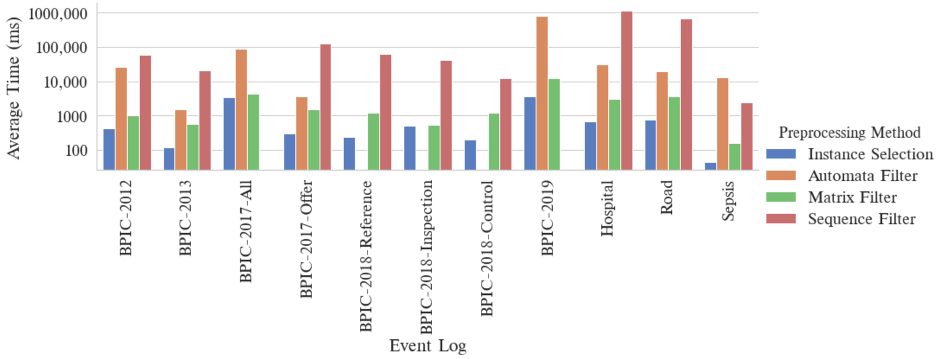


**Fig. 10.** The average of preprocessing time when we used instance selection strategies and three state-of-the-arts trace filtering methods [13, 18, 19].

presented in the event log. A low precision value means that the process model allows for much behavior compared to the given event log. There is a trade-off between these measures [50], sometimes, putting aside a small amount of behavior causes a slight decrease in the fitness value, whereas the precision value increases significantly. Therefore, we use the F-Measure metric that combines both of them with the same weight according to the following formula[11]:

$$F\text{-}Measure = \frac{2 \times Precision \times Fitness}{Precision + Fitness}. \qquad (8)$$

We did not consider the results of the Alpha miner algorithm as it may discover unsound process models, and consequently, it is not possible to compute the fitness value of them. Figure 11 compares the quality of best process models that are discovered with/without the instance selection method. We used sampled event logs just for discovery purpose, and the original event logs were used for computing F-Measure values. For the cases that the instance selection methods were used, we applied the sampled size in [1, 2, 3, 5, 10], and the average of F-Measure values is shown. For the statistical sampling method, we iterate the experiment four times, and again the average of F-Measure values are considered. For the *IMi* method, we used nine different filtering thresholds, but for other discovery methods, just the default setting was used. According to results that are presented in Figure 11, for the ILP, we always have an improvement when we use the instance selection method as a preprocessing step. However, the Split miner can usually discover process models with higher quality via the original event logs or using the statistical sampling method. Moreover, the statistical sampling method that randomly selects process instances results in process models with similar quality according to the F-Measure compared to original event logs. Note that instance selection strategies usually improve the quality when the size of the event log is considerable, e.g., *BPIC-2017-All* and *BPIC-2019*. In cases that the instance selection method could not improve the quality of process models, its quality is not much worse than using original event logs.

In Figure 12, the maximum F-Measure value of each instance selection strategy is shown using the previous setting. It is shown that for some event logs that there are

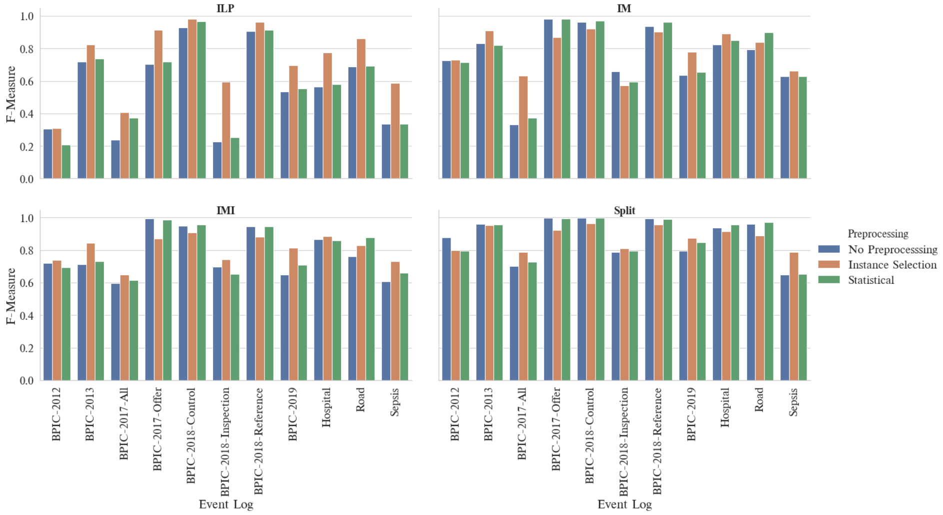


**Fig. 11.** Comparing the average F-Measure values of discovered process models with different subset selection methods.

high-frequent trace-variants, e.g., *Road* the frequency strategy outperforms other methods. Among ranking strategies, the longer strategy results in worse process models for most of the event logs.

For some event logs like *BPIC-2017-All*, the similarity-based selection strategy is the best choice for all of the process discovery algorithms. For the *Hospital* event log, the structure strategy usually results in process models with the highest F-Measure. Note that here, we show the maximum value of F-Measure, when we used different size of sampled event logs. Finding the best size and selection strategy is a challenging task, but as shown in this figure, we usually expect to have higher quality compared the original event log (i.e., ”*NO Preprocessing*”).

To understand a process model, the simplicity of it plays an important role. To measure the simplicity of a process model, we consider three metrics that measure the complexity of it. *Size* of process models is a combination of the number of transitions, places and arcs that connected them. Note that we used the Petri net notation for process models [35]. Another metric is the Cardoso metric [27] that measures the complexity of a process model by its complex structures, i.e., *And*, *Or*, and *Xor* components. Containing more complex structures requires much times to understanding behaviors, and consequently, it increases the complexity of the process model. There is also another metric called *Structuredness* that measures how different structural components in a Petri net is wrapped to each other [27]. It is expected that a Petri net with having more wrapped components is more difficult and requires much time to be understand. For all of these three measures, a lower value means less complexity and consequently a simpler process model.

In Table 5, we compared the complexity of process models when different preprocessing methods are applied. Here we used the same setting as explained in the previous experiment. For both the statistical and instance selection methods, we present the aver-

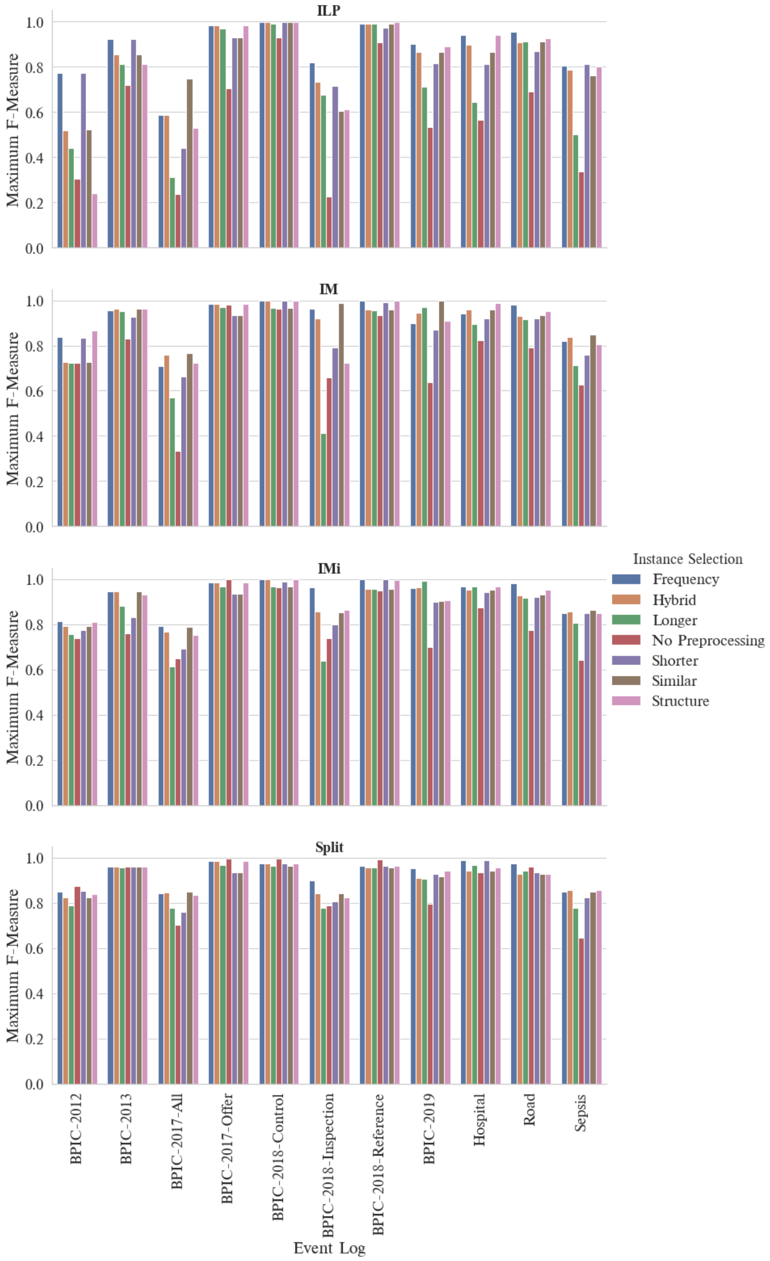
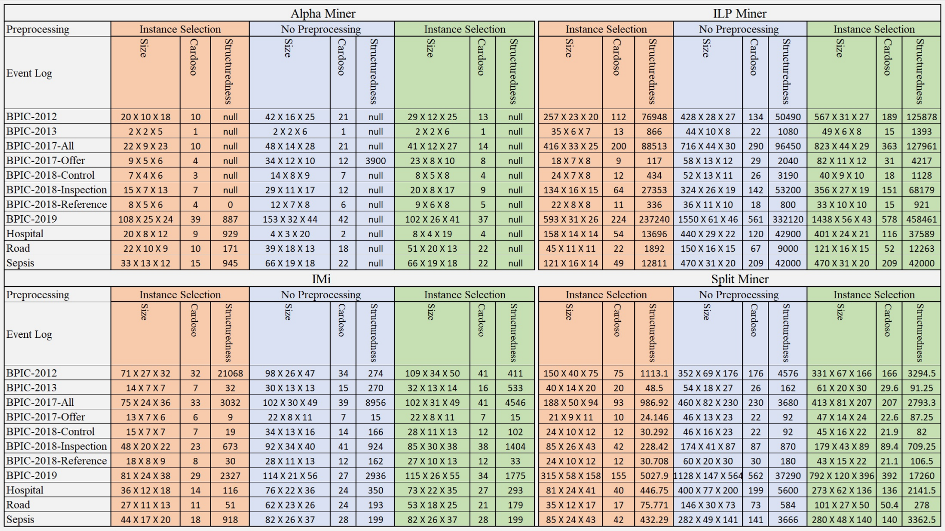


**Fig. 12.** The maximum F-Measure of discovered process models using different instance selection strategies.

**Table 5.** The average complexity metrics values of discovered process models using different process discovery algorithms. The size is representing the number of arcs, places, and transitions. The null values correspond to unsound process models.



| | Alpha Miner | | | | | | | | | ILP Miner | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Preprocessing | Instance Selection | | | No Preprocessing | | | Instance Selection | | | Instance Selection | | | No Preprocessing | | | Instance Selection | | |
| Event Log | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness |
| BPIC-2012 | 20 X 10 X 18 | 10 | null | 42 X 16 X 25 | 21 | null | 29 X 12 X 25 | 13 | null | 257 X 23 X 20 | 112 | 76948 | 428 X 28 X 27 | 134 | 50490 | 567 X 31 X 27 | 189 | 125878 |
| BPIC-2013 | 2 X 2 X 5 | 1 | null | 2 X 2 X 6 | 1 | null | 2 X 2 X 6 | 1 | null | 35 X 6 X 7 | 13 | 866 | 44 X 10 X 8 | 22 | 1080 | 49 X 6 X 8 | 15 | 1393 |
| BPIC-2017-All | 22 X 9 X 23 | 10 | null | 48 X 14 X 28 | 21 | null | 41 X 12 X 27 | 14 | null | 416 X 33 X 25 | 200 | 88513 | 716 X 44 X 30 | 290 | 96450 | 823 X 44 X 29 | 363 | 127961 |
| BPIC-2017-Offer | 9 X 5 X 6 | 4 | null | 34 X 12 X 10 | 12 | 3900 | 23 X 8 X 10 | 8 | null | 18 X 7 X 8 | 9 | 117 | 58 X 13 X 12 | 29 | 2040 | 82 X 11 X 12 | 31 | 4217 |
| BPIC-2018-Control | 7 X 4 X 6 | 3 | null | 14 X 8 X 9 | 7 | null | 8 X 5 X 8 | 4 | null | 24 X 7 X 8 | 12 | 434 | 52 X 13 X 11 | 26 | 3190 | 40 X 9 X 10 | 18 | 1128 |
| BPIC-2018-Inspection | 15 X 7 X 13 | 7 | null | 29 X 11 X 17 | 12 | null | 20 X 8 X 17 | 9 | null | 134 X 16 X 15 | 64 | 27353 | 324 X 26 X 19 | 142 | 53200 | 356 X 27 X 19 | 151 | 68179 |
| BPIC-2018-Reference | 8 X 5 X 6 | 4 | 0 | 12 X 7 X 8 | 6 | null | 9 X 6 X 8 | 5 | null | 22 X 8 X 8 | 11 | 336 | 36 X 11 X 10 | 18 | 800 | 33 X 10 X 10 | 15 | 921 |
| BPIC-2019 | 108 X 25 X 24 | 39 | 887 | 153 X 32 X 44 | 42 | null | 102 X 26 X 41 | 37 | null | 593 X 31 X 26 | 224 | 237240 | 1550 X 61 X 46 | 561 | 332120 | 1438 X 56 X 43 | 578 | 458461 |
| Hospital | 20 X 8 X 12 | 9 | 929 | 4 X 3 X 20 | 2 | null | 8 X 4 X 19 | 4 | null | 158 X 14 X 14 | 54 | 13696 | 440 X 29 X 22 | 120 | 42900 | 401 X 24 X 21 | 116 | 37589 |
| Road | 22 X 10 X 9 | 10 | 171 | 39 X 18 X 13 | 18 | null | 51 X 20 X 13 | 22 | null | 45 X 11 X 11 | 22 | 1892 | 150 X 16 X 15 | 67 | 9000 | 121 X 16 X 15 | 52 | 12263 |
| Sepsis | 33 X 13 X 12 | 15 | 945 | 66 X 19 X 18 | 22 | null | 66 X 19 X 18 | 22 | null | 121 X 16 X 14 | 49 | 12811 | 470 X 31 X 20 | 209 | 42000 | 470 X 31 X 20 | 209 | 42000 |

| | IMi | | | | | | | | | Split Miner | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Preprocessing | Instance Selection | | | No Preprocessing | | | Instance Selection | | | Instance Selection | | | No Preprocessing | | | Instance Selection | | |
| Event Log | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness | Size | Cardoso | Structuredness |
| BPIC-2012 | 71 X 27 X 32 | 32 | 21068 | 98 X 26 X 47 | 34 | 274 | 109 X 34 X 50 | 41 | 411 | 150 X 40 X 75 | 75 | 1113.1 | 352 X 69 X 176 | 176 | 4576 | 331 X 67 X 166 | 166 | 3294.5 |
| BPIC-2013 | 14 X 7 X 7 | 7 | 32 | 30 X 13 X 13 | 15 | 270 | 32 X 13 X 14 | 16 | 533 | 40 X 14 X 20 | 20 | 48.5 | 54 X 18 X 27 | 26 | 162 | 61 X 20 X 30 | 29.6 | 91.25 |
| BPIC-2017-All | 75 X 24 X 36 | 33 | 3032 | 102 X 30 X 49 | 39 | 8956 | 102 X 31 X 49 | 41 | 4546 | 188 X 50 X 94 | 93 | 986.92 | 460 X 82 X 230 | 230 | 3680 | 413 X 81 X 207 | 207 | 2793.3 |
| BPIC-2017-Offer | 13 X 7 X 6 | 6 | 9 | 22 X 8 X 11 | 7 | 15 | 22 X 8 X 11 | 7 | 15 | 21 X 9 X 11 | 10 | 24.146 | 46 X 13 X 23 | 22 | 92 | 47 X 14 X 24 | 22.6 | 87.25 |
| BPIC-2018-Control | 15 X 7 X 7 | 7 | 19 | 34 X 13 X 16 | 14 | 166 | 28 X 11 X 13 | 12 | 102 | 24 X 10 X 12 | 12 | 30.292 | 46 X 16 X 23 | 22 | 92 | 45 X 16 X 22 | 21.9 | 82 |
| BPIC-2018-Inspection | 48 X 20 X 22 | 23 | 673 | 92 X 34 X 40 | 41 | 924 | 85 X 30 X 38 | 38 | 1404 | 85 X 26 X 43 | 42 | 228.42 | 174 X 41 X 87 | 87 | 870 | 179 X 43 X 89 | 89.4 | 709.25 |
| BPIC-2018-Reference | 18 X 8 X 9 | 8 | 30 | 28 X 11 X 13 | 12 | 162 | 27 X 10 X 13 | 12 | 33 | 24 X 10 X 12 | 12 | 30.708 | 60 X 20 X 30 | 30 | 180 | 43 X 15 X 22 | 21.1 | 106.5 |
| BPIC-2019 | 81 X 24 X 38 | 29 | 2327 | 114 X 21 X 56 | 27 | 2936 | 115 X 26 X 55 | 34 | 1775 | 315 X 58 X 158 | 155 | 5027.9 | 1128 X 147 X 564 | 562 | 37290 | 792 X 120 X 396 | 392 | 17260 |
| Hospital | 36 X 12 X 18 | 14 | 116 | 76 X 22 X 36 | 24 | 350 | 73 X 22 X 35 | 27 | 293 | 81 X 24 X 41 | 40 | 446.75 | 400 X 77 X 200 | 199 | 5600 | 273 X 62 X 136 | 136 | 2141.5 |
| Road | 27 X 11 X 13 | 11 | 51 | 62 X 23 X 26 | 24 | 193 | 53 X 18 X 25 | 21 | 179 | 35 X 12 X 17 | 17 | 75.771 | 146 X 30 X 73 | 73 | 584 | 101 X 27 X 50 | 50.4 | 278 |
| Sepsis | 44 X 17 X 20 | 18 | 918 | 82 X 26 X 37 | 28 | 199 | 82 X 26 X 37 | 28 | 199 | 85 X 24 X 43 | 42 | 432.29 | 282 X 49 X 141 | 141 | 3666 | 280 X 48 X 140 | 140 | 3362.5 |

age values. In this table, *size* represents the number of elements in the discovered Petri net as $|A| \times |P| \times |T|$. Results show that using instance selection methods we will decrease the size of elements in discovered process models. Moreover, according to Cardoso values, the number of the complex component will be reduced when we use both preprocessing methods. However, using the instance selection method, the decrement is higher. In this table, there are some null values in the Structuredness field that indicate the corresponding process models are unsound. Therefore, as it is expected, the Alpha miner is not able to guarantee the soundness of discovered process models. Results illustrate that using instance selection strategies, for most of the cases, we are able to decrease the structuredness value or have simpler structures. In general, the reduction in complexity of discovered process models is higher for the Split miner that discovers more complex process models.

The average Cardoso values of different instance selection strategies are presented in Figure 13. It is shown that for most of the cases, all the instance selection strategies are having lower Cardoso values compared to the statistical sampling method. Among the instance selection methods, there is no method that dominantly outperforms others, however, the shorter strategy performs better than others. Except for the Alpha miner, the longer strategy usually results in process models with higher Cardoso value.

## 6. Conclusion

In this paper, we proposed some instance selection strategies to increase the performance of process discovery algorithms. We recommend applying process discovery algorithms on sampled event logs instead of whole event logs when dealing with large data sets. We

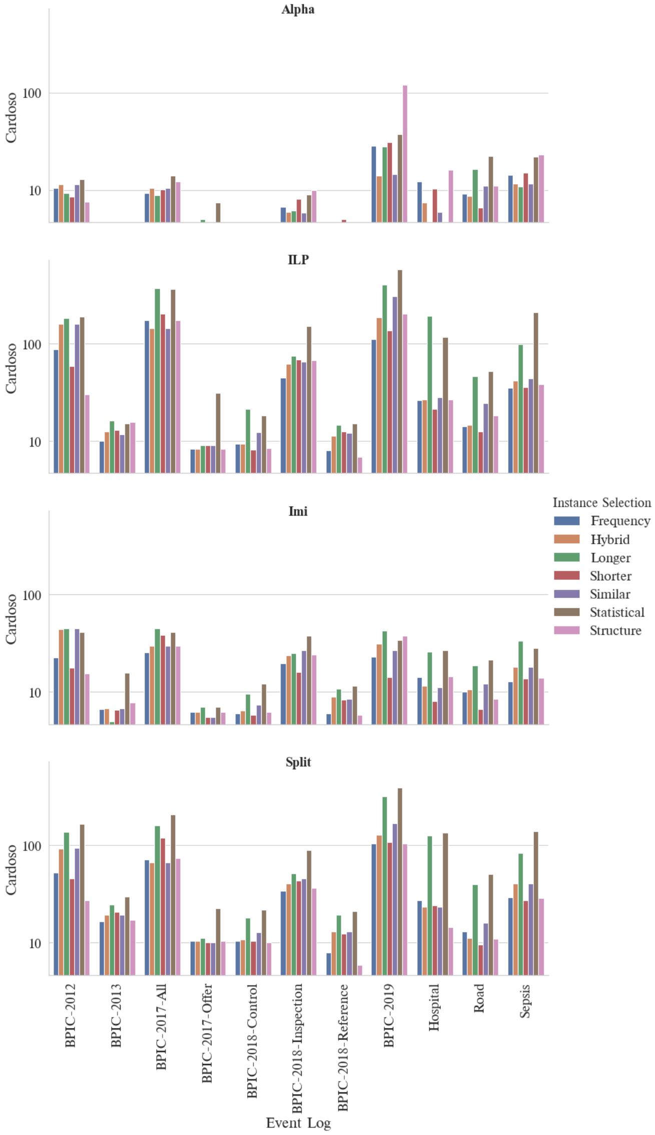


**Fig. 13.** The average Cardoso values of discovered process models using different preprocessing methods.

implemented different instance selection strategies in the ProM platform and also ported them into RapidProM.

To evaluate the proposed method, we applied it to several real publicly available event logs with different state-of-the-art process discovery algorithms. Experimental results show that by instance selection strategies, we are able to decrease the required time used by the process discovery algorithms. We found that instance selection strategies mostly increase the performance of process discovery by reducing the amount of behavior and the number of traces. Therefore, by using the sampled event logs, we are able to discover an acceptable approximation of the final process model in a shorter time. Using the sampled event log also results in having simpler process models according to complexity metrics. Moreover, the experiments indicate that for some event logs, instances selection strategies improve the quality of discovered process models according to the F-Measure metric for most of process discovery algorithms. The performance and F-Measure improvements for split miner are less than other process discovery algorithms. However, the instance selection strategies improve the complexity of this algorithm more than other methods. We found that the frequency, similarity and structured-based selection strategies result in process models with higher quality.

As future work, we aim to find out for different event logs and process discovery algorithms, which sampling strategy returns the best process model in less time. It is valuable to analyze the monotonicity of instance selection strategies and if possible guarantee specific requirements such as fitness or F-measure with the minimums size of the sampled event data.

**Acknowledgments.** We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research.

## References

1. van der Aalst, W.M.P.: Process Mining - Data Science in Action, Second Edition. Springer Berlin Heidelberg (2016)
2. van der Aalst, W.M.P., et all: Process mining manifesto. In: Business Process Management BPM Workshops, Clermont-Ferrand, France. pp. 169–194 (2011)
3. van der Aalst, W.M.P., Bolt, A., van Zelst, S.: RapidProM: Mine Your Processes and Not Just Your Data. CoRR abs/1703.03740 (2017)
4. van der Aalst, W.M.P., van Dongen, B., Günther, C.W., Rozinat, A., Verbeek, E., Weijters, T.: ProM: The Process Mining Toolkit. BPM (Demos) 489(31) (2009)
5. van der Aalst, W.M., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Trans. Knowl. Data Eng. 16(9), 1128–1142 (2004), https://doi.org/10.1109/TKDE.2004.47
6. Andrews, R., Suriadi, S., Ouyang, C., Poppe, E.: Towards event log querying for data quality - let's start with detecting log imperfections 11229, 116–134 (2018), https://doi.org/10.1007/978-3-030-02610-3_7
7. Augusto, A., Conforti, R., Dumas, M., Rosa, M.L., Polyvyanyy, A.: Split miner: automated discovery of accurate and simple business process models from event logs. Knowl. Inf. Syst. 59(2), 251–284 (2019), https://doi.org/10.1007/s10115-018-1214-x
8. Bauer, M., Senderovich, A., Gal, A., Grunske, L., Weidlich, M.: How much event data is enough? A statistical framework for process discovery. In: Krogstie, J., Reijers, H.A. (eds.) Advanced Information Systems Engineering - 30th International Conference, CAiSE 2018,

Tallinn, Estonia, June 11-15, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10816, pp. 239–256. Springer (2018), https://doi.org/10.1007/978-3-319-91563-0_15

9. Berti, A.: Statistical sampling in process mining discovery. In: The 9th International Conference on Information, Process, and Knowledge Management. pp. 41–43 (2017)
10. Boltenhagen, M., Chatain, T., Carmona, J.: Generalized alignment-based trace clustering of process behavior. In: Donatelli, S., Haar, S. (eds.) Application and Theory of Petri Nets and Concurrency - 40th International Conference, PETRI NETS 2019, Aachen, Germany, June 23-28, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11522, pp. 237–257. Springer (2019), https://doi.org/10.1007/978-3-030-21571-2_14
11. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: Meersman, R., Panetto, H., Dillon, T.S., Rinderle-Ma, S., Dadam, P., Zhou, X., Pearson, S., Ferscha, A., Bergamaschi, S., Cruz, I.F. (eds.) On the Move to Meaningful Internet Systems: OTM 2012, Confederated International Conferences: CoopIS, DOA-SVI, and ODBASE 2012, Rome, Italy, September 10-14, 2012. Proceedings, Part I. Lecture Notes in Computer Science, vol. 7565, pp. 305–322. Springer (2012), https://doi.org/10.1007/978-3-642-33606-5_19
12. Carmona, J., Cortadella, J.: Process mining meets abstract interpretation. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part I. Lecture Notes in Computer Science, vol. 6321, pp. 184–199. Springer (2010), https://doi.org/10.1007/978-3-642-15880-3_18
13. Conforti, R., Rosa, M.L., ter Hofstede, A.H.M.: Filtering out infrequent behavior from business process event logs. IEEE Trans. Knowl. Data Eng. 29(2), 300–314 (2017), https://doi.org/10.1109/TKDE.2016.2614680
14. De Leoni, M., Mannhardt, F.: Road traffic fine management process (2015), https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5
15. van Dongen, B., Borchert, F.: BPIC 2018. Eindhoven University of Technology (2018), https://doi.org/10.4121/uuid:3301445f-95e8-4ff0-98a4-901f1f204972
16. van Dongen, B.F., van der Aalst, W.M.P.: A meta model for process mining data 160 (2005), http://ceur-ws.org/Vol-160/paper11.pdf
17. Fani Sani, M., Berti, A., van Zelst, S.J., van der Aalst, W.M.P.: Filtering toolkit: Interactively filter event logs to improve the quality of discovered models 2420, 134–138 (2019), http://ceur-ws.org/Vol-2420/paperDT4.pdf
18. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Improving process discovery results by filtering outliers using conditional behavioural probabilities. In: Teniente, E., Weidlich, M. (eds.) Business Process Management Workshops - BPM 2017 International Workshops, Barcelona, Spain, September 10-11, 2017, Revised Papers. Lecture Notes in Business Information Processing, vol. 308, pp. 216–229. Springer (2017), https://doi.org/10.1007/978-3-319-74030-0_16
19. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Applying sequence mining for outlier detection in process mining. In: Panetto, H., Debruyne, C., Proper, H.A., Ardagna, C.A., Roman, D., Meersman, R. (eds.) On the Move to Meaningful Internet Systems. OTM 2018 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Malta, October 22-26, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11230, pp. 98–116. Springer (2018), https://doi.org/10.1007/978-3-030-02671-4_6
20. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Repairing outlier behaviour in event logs. In: Abramowicz, W., Paschke, A. (eds.) Business Information Systems - 21st International Conference, BIS 2018, Berlin, Germany, July 18-20, 2018, Proceedings. Lecture Notes in Business Information Processing, vol. 320, pp. 115–131. Springer (2018), https://doi.org/10.1007/978-3-319-93931-5_9
21. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Repairing outlier behaviour in event logs using contextual behaviour. vol. 14, pp. 5:1–5:24 (2018), https://doi.org/10.18417/emisa.14.5

22. Fani Sani, M., van Zelst, S.J., van der Aalst, W.M.P.: Conformance checking approximation using subset selection and edit distance. In: Dustdar, S., Yu, E., Salinesi, C., Rieu, D., Pant, V. (eds.) Advanced Information Systems Engineering - 32nd International Conference, CAiSE 2020, Grenoble, France, June 8-12, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12127, pp. 234–251. Springer (2020), https://doi.org/10.1007/978-3-030-49435-3_15
23. Fernández-Cerero, D., Varela-Vaca, Á.J., Fernández-Montes, A., López, M.T.G., Alvárez-Bermejo, J.A.: Measuring data-centre workflows complexity through process mining: the google cluster case. J. Supercomput. 76(4), 2449–2478 (2020), https://doi.org/10.1007/s11227-019-02996-2
24. Gao, J., van Zelst, S.J., Lu, X., van der Aalst, W.M.P.: Automated robotic process automation: A self-learning approach. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems: OTM 2019 Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC 2019, Rhodes, Greece, October 21-25, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11877, pp. 95–112. Springer (2019), https://doi.org/10.1007/978-3-030-33246-4_6
25. Günther, C.W., van der Aalst, W.M.P.: Fuzzy mining - adaptive process simplification based on multi-perspective metrics. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) Business Process Management, 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007, Proceedings, Lecture Notes in Computer Science, vol. 4714, pp. 328–343. Springer (2007), https://doi.org/10.1007/978-3-540-75183-0_24
26. Knols, B., van der Werf, J.M.E.M.: Measuring the behavioral quality of log sampling. In: International Conference on Process Mining, ICPM 2019, Aachen, Germany, June 24-26, 2019. pp. 97–104. IEEE (2019), https://doi.org/10.1109/ICPM.2019.00024
27. Lassen, K.B., van der Aalst, W.M.P.: Complexity metrics for workflow nets. Inf. Softw. Technol. 51(3), 610–626 (2009), https://doi.org/10.1016/j.infsof.2008.08.005
28. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - A constructive approach. In: Colom, J.M., Desel, J. (eds.) Application and Theory of Petri Nets and Concurrency - 34th International Conference, PETRI NETS 2013, Milan, Italy, June 24-28, 2013. Proceedings, Lecture Notes in Computer Science, vol. 7927, pp. 311–329. Springer (2013), https://doi.org/10.1007/978-3-642-38697-8_17
29. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs containing infrequent behaviour. In: Lohmann, N., Song, M., Wohed, P. (eds.) Business Process Management Workshops - BPM 2013 International Workshops, Beijing, China, August 26, 2013, Revised Papers, Lecture Notes in Business Information Processing, vol. 171, pp. 66–78. Springer (2013), https://doi.org/10.1007/978-3-319-06257-0_6
30. de Leoni, M., van der Aalst, W.M.P., Dees, M.: A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. Inf. Syst. 56, 235–257 (2016), https://doi.org/10.1016/j.is.2015.07.003
31. Luengo, D., Sepúlveda, M.: Applying clustering in process mining to find different versions of a business process that changes over time. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) Business Process Management Workshops - BPM 2011 International Workshops, Clermont-Ferrand, France, August 29, 2011, Revised Selected Papers, Part I. Lecture Notes in Business Information Processing, vol. 99, pp. 153–158. Springer (2011), https://doi.org/10.1007/978-3-642-28108-2_15
32. Mannhardt, F.: Hospital billing-event log. Eindhoven University of Technology. Dataset pp. 326–347 (2017)
33. Mannhardt, F., de Leoni, M., Reijers, H.A., van der Aalst, W.M.P.: Data-driven process discovery - revealing conditional infrequent behavior from event logs. In: Dubois, E., Pohl, K. (eds.) Advanced Information Systems Engineering - 29th International Conference, CAiSE 2017, Essen, Germany, June 12-16, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10253, pp. 545–560. Springer (2017), https://doi.org/10.1007/978-3-319-59536-8_34

34. Mendling, J.: Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness, Lecture Notes in Business Information Processing, vol. 6. Springer (2008), https://doi.org/10.1007/978-3-540-89224-3
35. Murata, T.: Petri nets: Properties, analysis and applications. Proceedings of the IEEE 77(4), 541–580 (1989)
36. Pegoraro, M., Uysal, M.S., van der Aalst, W.M.P.: Discovering process models from uncertain event data. In: Francescomarino, C.D., Dijkman, R.M., Zdun, U. (eds.) Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers. Lecture Notes in Business Information Processing, vol. 362, pp. 238–249. Springer (2019), https://doi.org/10.1007/978-3-030-37453-2_20
37. Pérez-Castillo, R., Fernández-Ropero, M., Piattini, M.: Business process model refactoring applying IBUPROFEN. an industrial evaluation. J. Syst. Softw. 147, 86–103 (2019), https://doi.org/10.1016/j.jss.2018.10.012
38. Pourbafrani, M., van Zelst, S.J., van der Aalst, W.M.P.: Scenario-based prediction of business processes using system dynamics. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems: OTM 2019 Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC 2019, Rhodes, Greece, October 21-25, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11877, pp. 422–439. Springer (2019), https://doi.org/10.1007/978-3-030-33246-4_27
39. Qafari, M.S., van der Aalst, W.M.P.: Fairness-aware process mining. In: Panetto, H., Debruyne, C., Hepp, M., Lewis, D., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems: OTM 2019 Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC 2019, Rhodes, Greece, October 21-25, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11877, pp. 182–192. Springer (2019), https://doi.org/10.1007/978-3-030-33246-4_11
40. Rafiei, M., van der Aalst, W.M.P.: Mining roles from event logs while preserving privacy. In: Business Process Management Workshops - BPM 2019 International Workshops, Vienna, Austria, September 1-6, 2019, Revised Selected Papers. pp. 676–689 (2019), https://doi.org/10.1007/978-3-030-37453-2_54
41. Sani, M.F., van Zelst, S.J., van der Aalst, W.M.P.: The impact of event log subset selection on the performance of process discovery algorithms. In: Welzer, T., Eder, J., Podgorelec, V., Wrembel, R., Ivanovic, M., Gamper, J., Morzy, M., Tzouramanis, T., Darmont, J., Latific, A.K. (eds.) New Trends in Databases and Information Systems, ADBIS 2019 Short Papers, Workshops BBIGAP, QAUCA, SemBDM, SIMPDA, M2P, MADEISD, and Doctoral Consortium, Bled, Slovenia, September 8-11, 2019, Proceedings. Communications in Computer and Information Science, vol. 1064, pp. 391–404. Springer (2019), https://doi.org/10.1007/978-3-030-30278-8_39
42. Song, M., Günther, C.W., van der Aalst, W.M.P.: Trace clustering in process mining. In: Ardagna, D., Mecella, M., Yang, J. (eds.) Business Process Management Workshops, BPM 2008 International Workshops, Milano, Italy, September 1-4, 2008. Revised Papers. Lecture Notes in Business Information Processing, vol. 17, pp. 109–120. Springer (2008), https://doi.org/10.1007/978-3-642-00328-8_11
43. van Zelst, S., van Dongen, B., van der Aalst, W.M.P., Verbeek, H.M.W.: Discovering Workflow Nets Using Integer Linear Programming. Computing (Nov 2017), https://doi.org/10.1007/s00607-017-0582-5
44. Suriadi, S., Andrews, R., ter Hofstede, A.H.M., Wynn, M.T.: Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. Inf. Syst. 64, 132–150 (2017), https://doi.org/10.1016/j.is.2016.07.011
45. van Dongen, B.F.: BPIC 2012. Eindhoven University of Technology (2012), https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
46. van Dongen, B.F.: BPIC 2017. Eindhoven University of Technology (2017), https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b

47. van Dongen, B.F.: BPIC 2019. Eindhoven University of Technology (2019), https://doi.org/10.4121/uuid:d06aff4b-79f0-45e6-8ec8-e19730c248f1
48. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Xes, xesame, and prom 6. In: Soffer, P., Proper, E. (eds.) Information Systems Evolution - CAiSE Forum 2010, Hammamet, Tunisia, June 7-9, 2010, Selected Extended Papers. Lecture Notes in Business Information Processing, vol. 72, pp. 60–75. Springer (2010), https://doi.org/10.1007/978-3-642-17722-4_5
49. Ward Steeman: BPIC 2013. Eindhoven University of Technology (2013), https://doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07
50. Weerdt, J.D., Backer, M.D., Vanthienen, J., Baesens, B.: A robust f-measure for evaluating discovered process models. In: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2011, part of the IEEE Symposium Series on Computational Intelligence 2011, April 11-15, 2011, Paris, France. pp. 148–155. IEEE (2011), https://doi.org/10.1109/CIDM.2011.5949428
51. Weerdt, J.D., vanden Broucke, S.K.L.M., Vanthienen, J., Baesens, B.: Active trace clustering for improved process discovery. IEEE Trans. Knowl. Data Eng. 25(12), 2708–2720 (2013), https://doi.org/10.1109/TKDE.2013.64
52. Weijters, A.J.M.M., Ribeiro, J.T.S.: Flexible heuristics miner (FHM). In: Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2011, part of the IEEE Symposium Series on Computational Intelligence 2011, April 11-15, 2011, Paris, France. pp. 310–317. IEEE (2011), https://doi.org/10.1109/CIDM.2011.5949453
53. van der Werf, J.M.E.M., van Dongen, B.F., Hurkens, C.A.J., Serebrenik, A.: Process discovery using integer linear programming. Fundam. Inform. 94(3-4), 387–412 (2009), https://doi.org/10.3233/FI-2009-136
54. van Zelst, S.J., van Dongen, B.F., van der Aalst, W.M.P.: Avoiding over-fitting in ilp-based process discovery. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) Business Process Management - 13th International Conference, BPM 2015, Innsbruck, Austria, August 31 - September 3, 2015, Proceedings, Lecture Notes in Computer Science, vol. 9253, pp. 163–171. Springer (2015), https://doi.org/10.1007/978-3-319-23063-4_10
55. van Zelst, S.J., Fani Sani, M., Ostovar, A., Conforti, R., Rosa, M.L.: Detection and removal of infrequent behavior from event streams of business processes. Inf. Syst. 90, 101451 (2020), https://doi.org/10.1016/j.is.2019.101451

**Mohammadreza Fani Sani** has started his Ph.D. at the Eindhoven University of Technology and continued it at RWTH Aachen University. His main research interest is in applying preprocessing techniques to improve performance and quality results of process mining algorithms.

**Sebastiaan J. van Zelst** received his Ph.D. degree in the area of online process mining from Eindhoven University of Technology. Currently, he is working as a scientist at the Fraunhofer Institute for Applied Information Technology (FIT), and he is a member of the Process and Data Science Chair of the RWTH Aachen University. His research interests include data-driven process optimization, process monitoring and prediction, event abstraction in process mining, and interactive process discovery.

**Prof.dr.ir. Wil van der Aalst** is a full professor at RWTH Aachen University leading the Process and Data Science (PADS) group. He is also part-time affiliated with the Fraunhofer-Institut für Angewandte Informationstechnik (FIT). In 2018, he was awarded an Alexander-von-Humboldt Professorship. His research interests include process mining, Petri nets, business process management, workflow management, process modeling, and process analysis. Wil van der Aalst has published over 800 articles, chapters, and books.

*Received: January 27, 2020; Accepted: September 1, 2020.*

Computer Science and Information Systems 17(3):959–982 https://doi.org/10.2298/CSIS200115029P

# Crowd counting á la Bourdieu *

## Automated estimation of the number of people

Karolina Przybylek[1] and Illia Shkroba[2] **

[1] University of Warsaw, Warsaw
00-721 Podchorazych 20, Poland
karolina.m.przybylek@gmail.com
[2] Polish-Japanese Academy of Computer Technology, Warsaw
02-008 Koszykowa 86, Poland
is@pjwstk.edu.pl

**Abstract.** In recent years, sociologists have taught us how important and emergent the problem of crowd counting is. They have recognised a variety of reasons for this fact, including: public safety (e.g. crushing between people, trampling underfoot, risk of spreading infectious disease, aggression), politics (e.g. police and government tend to underestimate the number of people, whilst protest organisers tend to overestimate it) and journalism (e.g. accuracy of the estimation of the ground truth supporting an article).
The aim of this paper is to investigate models for crowd counting that are inspired by the observations of famous sociologist Pierre Bourdieu. We show that despite the simplicity of the models, we can achieve competitive result. This makes them suitable for low computational power and energy efficient architectures.

**Keywords:** crowd counting, deep learning, mall dataset, habitus.

## 1. Introduction

Due to the increasing degree of urbanisation crowd management and control is a key issue for human life and security. For example, stampedes at the Kumbh Melas used to kill hundreds of people each event, until appropriate crowd management policy has been applied. In fact, many such problems related to crowds can be prevented or completely solved if aspects of crowd management and control are well organised. Crowd estimation is the first and foremost task in every crowd management process. Other crucial applications of crowd counting include politics and journalism: the number of people that take part in a given event indicates the strength and the impact of the event. Moreover, in our democratic world the number of people is the chief argument in every discussion. For this reason, crowd counts in presidential inauguration ceremonies (e.g. Obama in 2009 vs. Trump in 2017), demonstrations (e.g. "Black march" in 2018 in Poland) or festivals (e.g. LGBT Film Festival in 2011) are highly disputed. Automated crowd counting methods provide objective and indisputable estimation of the size of crowd.

* This paper is an extension of a paper initially published in ADBIS 2019 Workshops proceedings (Springer CCIS 1064).

** The names of the authors are arranged in alphabetical order.

The aim of crowd counting methods is to provide an accurate estimation of the number of people (the crowd) presented on a given picture. This task is extremely challenging for a number of reasons: scaling factor and perspective (people that are nearer to the camera appear much bigger), resolution of fragments containing a single person (i.e. few pixels per person), occlusion and clutter, illumination issues, distinct ambient environments, to name a few. Over last decades we witnessed a significant progress in automated crowd counting.

Two most successful classes of methods for crowd counting are: regression based and detection based methods. Regression based methods aim at estimating the density of the crowd by computing (either directly or indirectly) the heat-map of a picture. For example:

- A state-of-the-art regression model is described in [29].
- A very prominent approach to crowd counting by regression is described in [11]. The authors propose a multi-output regression model that learns how to balance local with global features (i.e. so called "spatially localised crowd counting").
- A variant of the previous method was proposed in [10]. The authors base their approach on the notion of "cumulative attributes", whose aim is to exploit deep correlations between different features.
- The first method capable of reliably counting thousands of people was proposed in [14]. The authors combine three sources of information to construct a regression model: Fourier transformation, interest points identification and head detection.
- A deep convolutional model to estimate the density map of an image is described in [5]. Actually, the main idea of the paper is to combine both deep and shallow convolutional neural networks to achieve good scaling effect.

On the other hand, detection based methods try to recognise some identifying features of each single person on a picture and sum up the number of recognised in this way people. For example:

- State-of-the-art pedestrian detection based on multiple features detection is described in [18]. The authors focuses on the problem of overlapping people as one of the main challenge to successful object-based detection. The method is optimised to count low density crowd and whose individuals have enough detail.
- The authors in [28] describe an end-to-end method of identification of pedestrians on images by using a novel architecture based on LSTM recurrent neural network. The results are competitive on TUDCrossing and Brainwash datasets. One may hope to further improve the method by using dynamic RNNs and Luong attention model as indicated in [7].
- It is well-known that naive transfer-learning does not work well in crowd counting. This issue is investigated in [30]. The authors proposes some methods to overcome such problem.

Our approach is a detection based method with explicit segmentation. We split an image into small segments, build a neural network to estimate the probability that a given segment contains a person, apply a cut-off point value and sum up all of the predictions. This approach was implicitly suggested by Pierre Bourdieu in [6], where he studied the concept of a *habitus*. The concept of habitus in terms of sociological theory of Pierre

Bourdieu is similar to the concept of personality. In fact, habitus could be defined as “social personality” as it especially focuses on the social aspects of the formation of personality. Like personality in psychological terms, habitus is a permanent “generative structure” that organises collective and individual experience. Moreover, it is also a structure that orients people’s attitudes and behaviours. There are different type of habitus — e.g. artistic habitus, legal habitus, prison habitus. The shape of these structures depends on the environmental conditions in which an individual functioned in the past and functions now. In other words, habitus depends on the social capital (including economic capital, cultural capital, symbolic capital, etc.). The sociologist argued that habitual complexity of individuals is present at different levels of scale, and, therefore, one may identify a person (or even the social class of a person) by looking only at some of her tiny details in isolation. We show that the models that we build, despite their simplicity, are competitive on Mall Dataset [1] (also see: [9], [10], [19] and [11]). The simplicity of our models makes them suitable for mobile and embedded architectures, where computational power and space are limited. We note that other methods inspired by social observations were successfully used in the past, for example: [4] and its extensions [23], [22].

The source code of our methods is available at a public repository:

https://github.com/s14028/engineer

The paper is the extension of paper [21] initially published in ADBIS 2019 Workshops proceedings. The organisation of the paper is as follows. In the next section we describe the mathematical model behind our estimation method. The section covers the following concepts: the basic idea of a neural network, together with convolutional neural networks, binary cross-entropy loss function, max pooling, and common problems that one may encounter while training neural networks, or selecting models. The experimental environment is described in Section 3. The dataset is described in Subsection 3.1 and our transformations of datasets together with necessary preprocessing (i.e. data segmentation) in the following subsection. Subsection 3.3 describes a perspective map. Subsection 3.4 describes our approach to image augmentation. Subsection 3.5 describes the hardware that we used to train our models. Subsection 3.6 describes the error metrics that we use to assess the quality of the models. In Section 4 we describe three models. the first one is described in Subsection 4.1 and operates on raw images, the second of the models, described in Subsection 4.2, utilises additional information from the perspective map. These two models use the standard binary cut-off point method to decide whether a given segment of a picture contains a person. The third model, described in Subsection 4.3, also utilises additional information from the perspective map, but uses a more advanced continuous cut-off point method. We summarise the results and conclude the paper in Section 5.

## 2. Neural Networks

As mentioned in the introduction, all our models are represented as, so called, neural networks. A neural network is a mathematical model of computation based on layers of neurons connected together. The aim of this section is to provide basic information about neural networks that are necessary to understand and replicate our results. We will begin this section with the explanation of a model of a single neuron. Then we tell how multiple neurons can be connected to form a layer. Finally, we describe how multiple layers can be stacked on top of each other to form a neural network. We will also describe some common pitfalls in choosing an appropriate model and in training the models.

### 2.1. Artificial neuron

A neuron in a neural network is a function $f\colon \mathbb{R}^k \mapsto \mathbb{R}$ of the form as in Equation (1), where $w \in \mathbb{R}^k$ is a vector called the *weights* for neuron $f$ :

$$f(x_1, x_2, \ldots, x_k) = \sum_{j=1}^{k} w_j x_j \tag{1}$$

The function $f$ is be usually written in a "dot product" notation as shown in Equation (2).

$$w \cdot x = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix} = w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_k \cdot x_k = f(x_1, x_2, \ldots, x_k) \tag{2}$$

Groups of neurons in a neural network are arranged together in "layers". A layer takes some input vector $x$ and forwards it to each of its neurons. The outputs of the neurons are connected together to form a vector $y$. If a layer has $n$ neurons, vector $y$ has dimension $n$. Then, vector $y$ is passed to an "activation function" $g\colon \mathbb{R}^n \mapsto \mathbb{R}^n$. A layer can be represented as a pair: a matrix $A$ and an activation function $g$, as shown in Equation (3), where each row represents a neuron in the layer.

$$A = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1k} \\ w_{21} & w_{22} & \cdots & w_{2k} \\ \vdots & \vdots & \cdots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nk} \end{bmatrix} \tag{3}$$

Therefore, a layer $\langle Ag \rangle$ linearly transforms vector $x$ to vector: $g(Ax)$ — i.e. we can think of a layer as a linear transformation composed with some non-linear activation function. If we combine several layers together, we will obtain a neural network. Of course, the dimensions of the consecutive layers have to agree. If one layer is represented as a matrix $A$ of dimension $n \times k$, the following layer represented as a matrix $B$ must have dimension $m \times n$. For example, if the layers have activation functions $g\colon \mathbb{R}^n \mapsto \mathbb{R}^n$ and $h\colon \mathbb{R}^m \mapsto \mathbb{R}^m$ respectively, we get the following network $M\colon \mathbb{R}^k \mapsto \mathbb{R}^m$, that is shown in Equation (4).

$$M(x) = h(B(g(Ax))) \tag{4}$$

### 2.2. Bias

A *bias* is an additional vector of weights $b \in \mathbb{R}^n$ that is added to the output vector $y \in \mathbb{R}^n$ of a layer just before applying the activation function $g\colon \mathbb{R}^n \mapsto \mathbb{R}^n$. This additional vector allows the network to “shift” vector $y$. Equation (5) shows the transformation made by a layer with a bias.

$$g(y+b) = g(Ax+b) \tag{5}$$

Which is equivalent to Equation (6).

$$g(y') = g(A'x') \tag{6}$$

Where $A' \in \mathbb{R}^{n \times k+1}$ is presented in Equation (7).

$$A' = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1k} & b_1 \\ w_{21} & w_{22} & \cdots & w_{2k} & b_2 \\ \vdots & \vdots & \dots & \vdots & \vdots \\ w_{n1} & w_{n2} & \cdots & w_{nk} & b_n \end{bmatrix} = \begin{bmatrix} A \; b \end{bmatrix} \tag{7}$$

And where $x' \in \mathbb{R}^{k+1}$ is presented in Equation (8).

$$x' = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ 1 \end{bmatrix} \tag{8}$$

Before moving to a possible interpretation of the concept of bias, let us recall (Equation (9)) another representation for the “dot product” of vectors.

$$w \cdot x := ||w||||x|| \cos \alpha \tag{9}$$

Where $||-||$ denotes the length of a vector, and $\alpha$ is the angle between vectors $w$ and $x$. If the length of each vector is equal 1, then the dot product is equal to $\cos \alpha$. If the angle between $w$ and $x$ is approaching $0$, then the dot product is approaching $||w|| \cdot ||x|| \cdot 1$. If the angle between $w$ and $x$ is approaching $\pi$, then the dot product is approaching $||w|| \cdot ||x|| \cdot (-1)$. And if the angle between $w$ and $x$ is approaching $\frac{\pi}{2}$ or $\frac{3\pi}{2}$, then the dot product is approaching $||w|| \cdot ||x|| \cdot 0$.

Let us consider a vector $w \in \mathbb{R}^k$ and assume that $w_i \neq 0$ for some $i \leq k$. Then $w \cdot x$ gives the Equation (10).

$$x_j = -\frac{1}{w_j}(w_1 \cdot x_1 + \cdots + w_{j-1} \cdot x_{j-1} + w_{j+1} \cdot x_{j+1} + \ldots + w_k \cdot x_k) \tag{10}$$

Where $x_1, ..., x_{j-1}, x_{j+1}, ..., x_k \in \mathbb{R}$ are arbitrary real numbers, and $x_j$ can be calculated from the equation. This means, that vector $x$ could be represented by some vector $(x_1, ..., x_{j-1}, x_{j+1}, ..., x_k) \in \mathbb{R}^{k-1}$ together with Equation (10). The subspace $\mathbb{R}^{k-1}$ spanned by all such vectors is called the hyperplane of vector $w$ (it is the hyperplane orthognoal to $w$).

To get some intuitions about the concept of bias, let us consider the following example. Suppose that we are dealing with a single neuron represented as function $f \colon \mathbb{R}^2 \mapsto \mathbb{R}$.

The input to the neuron consists of vectors in $\mathbb{R}^2$ that belong to two different classes: $\triangle$ or $\bigcirc$. For the vectors of class $\triangle$, the neuron has to assign a value $y \in (-\infty, 0)$, and for vectors of class $\bigcirc$, the neuron has to assign a value $y \in \langle 0, \infty)$ — i.e. the neuron has to separate the classes (this is a typical classification problem). Some sample input vectors are presented on Figure 1.

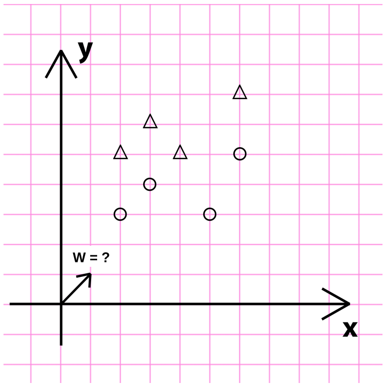


**Fig. 1.** Vectors of classes $\bigcirc$ and $\triangle$

The weight vector $w$ associated with a neuron for this classification problem, should have an angle greater than $\frac{\pi}{2}$ and smaller than $\frac{3\pi}{2}$ to classify $\triangle$, and should have an angle smaller or equal than $\frac{\pi}{2}$ or bigger or equal than $\frac{3\pi}{2}$ to classify $\bigcirc$. Unfortunately, no vector $w$ can satisfy these requirements.

Bias solves this problem, by extending the weight vector $w$ and the input vector $x$ by an additional dimension. Each input vector $x$ has assigned a constant value 1 in that dimension. This means that each of these vectors live in some $\mathbb{R}^2$ subspace of the extended $\mathbb{R}^3$ space. In contrast, weight vector $w$ has some value $b \in \mathbb{R}\colon b \neq 0$ in that dimension. This allows vector $w$ to have a hyperplane $\mathbb{R}^2$ to differentiate between input vectors.

Consider the intersection of the hyperplane of vector $w$ with the space spanned by the input vectors. Assume that $w$ is extended to an additional dimension with $w_{k+1} = b$ and $x$ with $x_{k+1} = 1$. For each vector $v \in \mathbb{R}^{k+1}$, from the hyperplane of vector $w$ , we can define the value $v_{k+1}$ (11) by using Equation (10).

$$v_{k+1} = -\frac{1}{b}(w_1 \cdot v_1 + w_2 \cdot v_2 + \cdots + w_k \cdot v_k) \tag{11}$$

To obtain the intersection we have to consider the situation when vectors $v$ and $x$ are equal. For some $i \in \{1, 2, \ldots, k\}$ the value $v_i \in \mathbb{R}$ could be equal to value $x_i \in \mathbb{R}$, thus find the intersection, we have to check when $v_{k+1} = 1$, as shown in Equation (12) and Equation (13).

$$-\frac{1}{b}(w_1 \cdot v_1 + w_2 \cdot v_2 + \cdots + w_k \cdot v_k) = 1 \tag{12}$$

$$-\frac{1}{w_j}(w_1 \cdot v_1 + \cdots + w_{j-1} \cdot v_{j-1} + w_{j+1} \cdot v_{j+1} + \cdots + w_k \cdot v_k + b) = v_j \tag{13}$$

Where $v_1, v_2, ..., v_{j-1}, v_{j+1}, ..., v_k \in \mathbb{R}$ could be any real numbers, and $v_j$ depends on $v_i$ for $i \neq j$. Therefore, that vector $v$ can be represented as $(v_1, v_2, ..., v_{j-1}, v_{j+1}, ..., v_k) \in \mathbb{R}^{k-1}$ and Equations (11) together with Equation (13). This shows that the intersection of the hyperplane of $w$ and the space spanned by the input vectors is a subspace $\mathbb{R}^{k-1}$. In our case, when $k = 2$, the intersection is a subspace $\mathbb{R}^1$, which means, that the intersection is just a line.

This leads us to the conclusion, that bias allows us to think of a neuron as a shifted hyperplane dividing the space spanned by the input vectors, as shown on Figure 2.

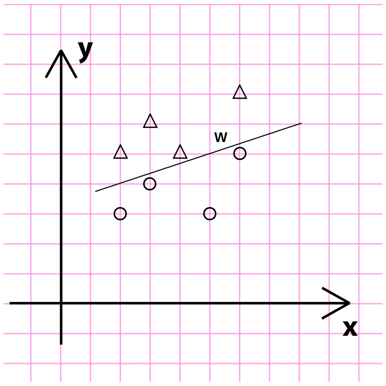


**Fig. 2.** Separating vectors for classes ◯ and △ with line

## 2.3. Binary Cross-Entropy Loss

In the process of training, we minimise a loss function $E$ associated with the model by adjusting the weights of the model. A single weight adjustment is called a "step". A single run through the whole training set is called an "epoch". An epoch consists of several steps. A set of inputs associated with the step is called a "batch", and length of these inputs set is called a "batch size".

For example, let us assume, that our training set $X$ along with its true labels $D$ looks as shown on Equation (14) and Equation (15):

$$X = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}, I_{11}, I_{12}, I_{13}\} \tag{14}$$

$$D = \{d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_9, d_{10}, d_{11}, d_{12}, d_{13}\} \tag{15}$$

For batch size $m = 2$, some batches are shown in Equation (16) and Equation (17).

$$X/\sim = \{\{I_1, I_2\}, \{I_3, I_4\}, \{I_5, I_6\}, \{I_7, I_8\}, \{I_9, I_{10}\}, \{I_{11}, I_{12}\}, \{I_{13}\}\} \tag{16}$$

$$D/\sim = \{\{d_1, d_2\}, \{d_3, d_4\}, \{d_5, d_6\}, \{d_7, d_8\}, \{d_9, d_{10}\}, \{d_{11}, d_{12}\}, \{d_{13}\}\} \tag{17}$$

For each of these batches, predictions will be made by the model and true labels along with predictions will be used for calculating a binary cross-etropy loss $E$. The Binary cross-entropy loss is widely used for training binary classifiers. Its definition is presented as Equation (18).

$$E(d, y) = -lr \sum_{i=1}^{n} d \cdot log_2(y) + (1-d) \cdot log_2(1-y) \tag{18}$$

Where $lr \in \mathbb{R}$ is a hyperparameter called the "learning rate". What distinguishes hyperparameter of a model from parameter, is that hyperparameter is not being trained. We have to tune a value of a hyperparameter ourselves. By tuning hyperparameter $lr$ we can control how fast loss function converges to minimum. If a learning rate is too high, model can bypass minimum of loss function. If a learning rate is too low, model will train slowly and could stuck in a local minimum.

### 2.4. The idea of Convolutional Neural Network

The convolutional neural network is a type of a neural network model widely used for image recognition. The reason for that, is that convolutional neural network is capable of learning and recognising "similar patterns" on far different parts of an image. Before we explain convolutional neural network, we have to recall the idea of the convolution in mathematics.

Let us assume, that we have a relay-race with 2 members. The whole distance of the race is equal to $t \in \mathbb{R}$. The function $f \colon \mathbb{R} \mapsto \langle 0, 1 \rangle$ is the probability of running some distance $x \in \mathbb{R}$ by the runner who starts the race. The covered distance $x$ could, of course, be smaller than the whole distance $t$. If the first runner stops at some point $x$ then the second runner must start from the same point $x$ and has to finish the race by running distance $t - x$ with probability function given by $g \colon \mathbb{R} \mapsto \langle 0, 1 \rangle$.

If we consider every possible position $x$, where the first runner stops, and calculate the probability of "winning" by reaching the finish line (i.e. covering the whole distance $t$), we will get the convolution of functions $f$ with $g$ as shown in Equation (19).

$$(f * g)(t) = \int_{-\infty}^{\infty} f(x) \cdot g(t-x) dx \tag{19}$$

The discrete version of the convolution for discrete probability functions has following definition (20):

$$(f * g)(t) = \sum_{x} f(x) \cdot g(t-x) \tag{20}$$

In convolutional neural network, the role of function $f$ is played by the input image, and the role of function $g$ is played by the kernel (also known as the filter). Assume, that the input image is represented as a matrix $I \in \mathbb{R}^{n \times m}$, and the kernel is a matrix $K \in \mathbb{R}^{r \times k}$, where $n, m \in \mathbb{N}$, and $r \leq n \wedge k \leq m$. The kernel, starting from the top left corner moves through entire $x$-axis with some stride until it reaches the right corner. After reaching the right corner, the kernel is placed at the beginning of $x$-axis, then slightly shifted down through $y$-axis with some stride, and the process repeats, until the kernel reaches the bottom right corner. During each shift, the convolution of the kernel with the covered part of the image is calculated, and the result is stored as the "output image" $O$.

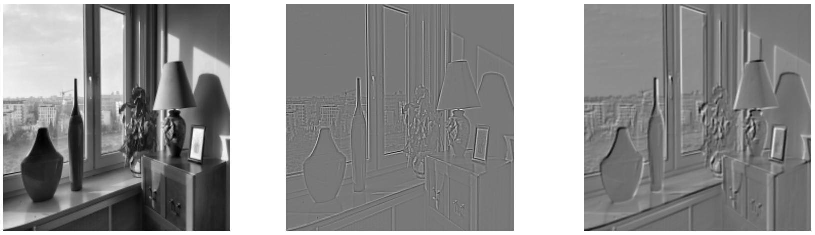
An example of convolution with stride $1 \times 1$ is shown in Equations (21), (22) and (23):

$$I = \begin{bmatrix} 5 & 3 & 2 \\ 1 & 5 & 6 \\ 3 & 4 & 1 \end{bmatrix} \quad (21) \qquad K = \begin{bmatrix} -2 & -1 \\ -1 & 1 \end{bmatrix} \quad (22)$$

$$O = \begin{bmatrix} (-2 \cdot 5 + -1 \cdot 3 + -1 \cdot 1 + 1 \cdot 5) & (-2 \cdot 3 + -1 \cdot 2 + -1 \cdot 5 + 1 \cdot 6) \\ (-2 \cdot 1 + -1 \cdot 5 + -1 \cdot 3 + 1 \cdot 4) & (-2 \cdot 5 + -1 \cdot 6 + -1 \cdot 4 + 1 \cdot 1) \end{bmatrix} \quad (23)$$

A layer in the convolutional neural network is represented by a sequence of kernels. Each kernel in the process of training learns to "highlight" different patterns on an image. For example, kernel $K_1$ (24) is capable of detecting edges and kernel $K_2$ (25) embosses image. The resulting convolutions are presented along with the original image on Figure 3.

$$K_1 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (24) \qquad K_2 = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix} \quad (25)$$



(a) Original image (b) edge detection with kernel $K_1$ (c) emboss with kernel $K_2$

**Fig. 3.** The result of the convolution process applied to image (a) with kernel $K_1$ (b) and $K_2$ (c)

The process of training of a convolutional neural network is a described in the above. The loss function is minimised and, consequently, the weights of the kernels are modified.

If an image consists of $c \in \mathbb{N}$ color channels, then the kernel is represented as $c$ matrices instead of a single one. Each of these kernels is applied accordingly to the color channels and the convolution is calculated. The resulting convolutions for each of the color channels are summed together into a single color channel of the resulting image.

Therefore, if the input image to the layer has *RGB* colors, the output image has as much color channels as the number of kernels.

### 2.5. Max Pooling

In order to reduce the memory consumption and the model complexity, *max pooling* operation is applied to the images. Max pooling splits its input image into small segments and substitutes these segments by the "representative pixels". The representative pixel is the maximum of the values of the pixels in a given segment. An example of max pooling is presented on Figure 4.

$$\underbrace{\begin{bmatrix}1&3&7&9\\5&2&1&7\\3&2&1&7\\2&3&7&5\end{bmatrix}}_{\text{Image}} \xrightarrow{\text{Max Pooling}} \begin{matrix}\begin{bmatrix}1&3\\5&2\end{bmatrix}&\begin{bmatrix}7&9\\1&7\end{bmatrix}\\\begin{bmatrix}3&2\\2&3\end{bmatrix}&\begin{bmatrix}1&7\\7&5\end{bmatrix}\end{matrix} \xrightarrow{\text{Result}} \begin{bmatrix}5&9\\3&7\end{bmatrix}$$

**Fig. 4.** An example of max pooling with segment shape $2 \times 2$

Max pooling also works with shapes of segments that cannot be fitted into the whole image. For example, if we split the original image $I \in \mathbb{R}^{5\times 5}$ into segments with the highest dimension $2 \times 2$, then $4$ segments will be created with dimension $2 \times 2$, 2 segments with dimension $2 \times 1$, 2 segments with dimension $1 \times 2$ and a single segment with dimension $1 \times 1$. Each of those segments will be substituted by its representative pixel. Further description of max pooling could be found in article [12].

### 2.6. Flatten

The flatten layer is used when a sub-neural network is stacked on top of a convolutional neural network to form a combined model. The sub-neural network takes a 1-dimensional vector $x \in \mathbb{R}^k$ as its input, however the convolutional network outputs a 3-dimensional matrix $O \in \mathbb{R}^{n\times m\times r}$. The flatten layer connects these two networks by flattening matrix $O$ to vector $x$.

### 2.7. Common pitfalls

In this subsection we describe some common problems that one may encounter when selecting models for a given problem, or when training the models. We give some insights how to solve or avoid the potential problems.

**Overfitting** When the loss function $E$ on the training set approaches zero, but on the validation and test sets is relatively high, a phenomenon called "model overfitting" appears. During the training process, the model starts to learn the noise that is present in the data and miss-interpret it as some important feature of the input. The overfitted model is a model with too high complexity for a given problem.

As an example let us consider a model that recognises different animals on images. Overfitted model could create association, that if some pixels on the image has some specific colors, then there is a polar bear on the image.

**Underfitting** An opposite phenomenon to the "model overfitting" can also occur. A model could be too simple to create some complicated associations and, consequently, learns overly generalised concepts. For example, an "underfitted model" can learn that if an image has a high white color balance, then we are dealing with polar bear on an image. This mistaken association happens, because images containing polar bears are often being made in snow environments, so the image mostly consists of white colors. Thereofre, an underfitted model can have problems in distinguishing a polar bear from a polar wolf.

**Regularisation** One way of dealing with overfitting is by using "regularisation" to reduce the complexity of the models. The complexity of a neural network grows with the increased number of weights and layers. Techniques used for the reduction of model complexity are called "regularisation techniques". In the below we will mention a couple of the most common regularisation techniques, especially focusing on these that we have used in our models.

*Dimensionality Reduction.* If we consider a vector $x \in \mathbb{R}^k$ as an input of a layer $A \in \mathbb{R}^{n \times k}$, the reduction of the dimension of vector $x$ will also reduce the number of columns in matrix $A$. In this way, the complexity of a neural network can be reduced. It is also possible to reduce the number of neurons in matrix $A$. Consider a neural network $M$ (26).

$$M(x) = h(B(g(Ax))) \tag{26}$$

If we reduce the number of neurons in layer $A$ to some value $t \in \mathbb{N} : 0 < t < n$, the number of columns of matrix $B$ will be accordingly reduced to $t$.

*Regularisation $l_1$ and $l_2$.* Often we do not know what is a good way of reducing the dimensionality of the input features and the number of neurons. To remove unnecessary complexity without modifying the structure of the model (i.e. without removing neurons or layers) we can substitute unnecessary weights with zeros. Unfortunately, we do not know which weights are important and which are not. The process of finding and removing unnecessary weights can be achieved by providing additional loss function to minimise the total size of the weights. More specifically, we can add a loss function whose aim is to reduce weights like on Equation (27), where $w \in \mathbb{R}^n :\, n \in \mathbb{N}$ is a sequence of the weights of the whole model and $\alpha \in \mathbb{R}$ is a hyperparameter.

$$l_1 = \alpha \sum_{i=1}^{n} |w_i| \tag{27}$$

Regularisation function $l_1$ is also commonly known as "Least Absolute Shrinkage and Selection Operator (LASSO)". Minimising several loss functions is called "multicriteria optimisation" and in our case is achieved by adding them together (28).

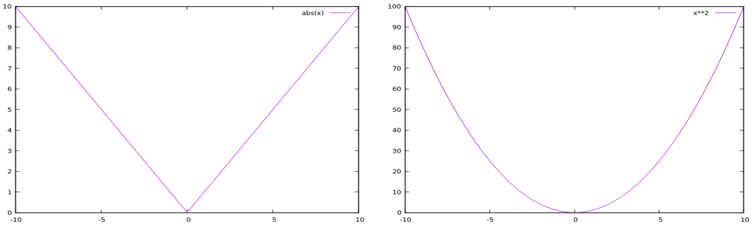
$$E' = E + l_1 \tag{28}$$

By tuning hyperparameter $\alpha$, we can control the level of regularisation. If the level of regularisation is too high, all weights of the model could reach values close to zero. If a regularisation level is too low, the model can still overfit.

A similar regularisation function $l_2$, also known as "ridge" is defined by Equation (29).

$$l_2 = \alpha \sum_{i=1}^{n} w_i^2 \tag{29}$$

Comparing function $l_1$ and $l_2$, we can observe that $l_1$ function is more "aggressive". By looking at the shapes of these functions (see Figure 5), we can see that $l_1$ has the same speed of convergence, independently of the values of the weights. In contrast, $l_2$ function converges slower, when the values of the weights are small.



(a) $l_1$ regularisation (b) $l_2$ regularisation

**Fig. 5.** Functions applied by regularisers $l_1$ and $l_2$

*Data Augmentation.* If an overfitted model creates artificial associations, we can provide to the model additional slightly augmented images. The reason behind data augmentation is that collecting enough training samples can be often problematic or even impossible. The idea of data augmentation is as follows: instead of collecting new training samples, the samples are generated as a slight modification of the current training samples. Some of the possible modifications are:

- rotation
- shift by $x$ axis
- shift by $y$ axis
- vertical flip
- horizontal flip
- shear
- zoom

- change of the color map

Our method of image augmentation is based on rotations with filling segment gaps by segment surrounding and is described in details in Section 3.4.

**Balance between Overfitting and Underfitting** Sometimes reducing the complexity of a model can lead to underfitting. Dimensionality reduction may remove some important features from the data that cannot be easily noticed or seems to be unimportant only in a specific training samples. A neural network with some neurons and layers removed may not be able to learn complicated patterns in the data. Finally, after providing new samples with data augmentation it may be too difficult for a model to learn other associations than the primitive ones.

When a model starts to underfit after applying complexity reduction, we have to increase its complexity, by adding additional layers, neurons or reducing the level of regularisation. After increasing the complexity of the model, we can face again the problem of overfitting. Such a situation leads to a loop called "balance between overfitting and underfitting", which is explained in article [2]. There is no simple solution to this problem. During our research, we used the following technique. We started with building a simple model with only two layers — one hidden layer and the ouptut layer, and used this model as the "baseline model". This baseline model gave some results. Then we applied regularisation to it. After applying the regularisation the model gave worse results, so we significantly increased model complexity, by adding more layers and neurons. This model started to overfit quickly, so we applied a higher level of regularisation (i.e. we used $\alpha = 0.001$ for $l_1$ regulariser and added data augmentation). In the result, the model started to underfit, so we simply lowered the regularisation hyperparameters obtaining our final results.

## 3. Experimental Environment

We evaluate our models and compare against the state-of-the-art model on Mall Dataset from [1].

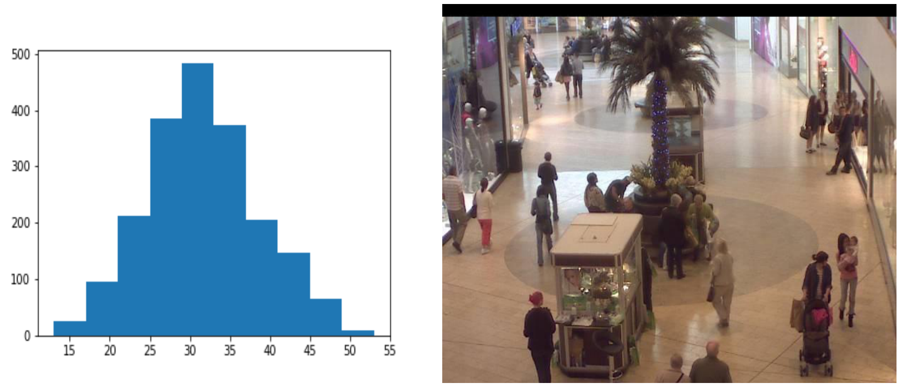
### 3.1. Dataset

The data from Mall Dataset consists of:

- 2000 images taken from the same camera with the same perspective and with resolution $640 \times 480$ pixels each;
- the coordinates of people's faces on images;
- the perspective map of the images (see Subsection 3.3).

The distribution of the number of people in the database is shown on Figure 6(a). The distribution clearly resembles the Gaussian distribution with mean $31.16$ and standard deviation $6.94$. Basic statistics for the whole dataset are as follows:

- maximum number of people on an image: $53$;
- minimum number of people on an image: $13$;



(a) Distribution of crowd sizes across samples

(b) Sample image

**Fig. 6.** Distribution of crowd sizes along with sample image from Mall dataset

- average number of people on an image: $31.16$;
- median number of people on an image: $31$;
- standard deviation of the number of people on an image: $6.94$.

A sample image from the dataset is shown on Figure 6(b).

The dataset was randomly split into:

- the training set consisting of $1500$ images;
- the validation set consisting of $300$ images;
- the test set consisting of $200$ images.

### 3.2. Data Segmentation

Since we were interested in low memory cost of crowd counting, we could not work with full images. We decided to split each image into smaller segments of the same shapes and with each segment associate a binary value indicating whether or not a person is present on the segment. We accomplished this by creating a map that to the coordinate of the centre of a person's face assigns the segment that contains it. In this process we lost some information, but the accuracy of the final classifiers shows that the loss was not that big.

We examined several splitting methods: we split images into 100, 400 and 1600 non-overlapping rectangular segments of the same shape[3]. Figure 7 shows segments of an image after splitting it on a various number of segments. It is worth noticing, that a person's sweater is barely visible when we split the image on 1600 segments. We have found that splitting into 400 segments performs best for our models. One may argue that this splitting method has a reasonable ratio of segments with a person to segments without a person at around $0.0679$, and at the same time it has a decently low loss rate. Table 1 summarises the statistics for other splitting methods.

[3] Notice, that we did not use content-based image segmentation.

**Table 1.** Statistics for different number of segments for image splitting

| Segments | Ratio | Mean loss | Std loss |
|---|---|---|---|
| 100 | 0.2105 | −10.1035 | 3.7835 |
| 400 | 0.0679 | −3.9995 | 2.1153 |
| 1600 | 0.0188 | −1.0040 | 1.0144 |

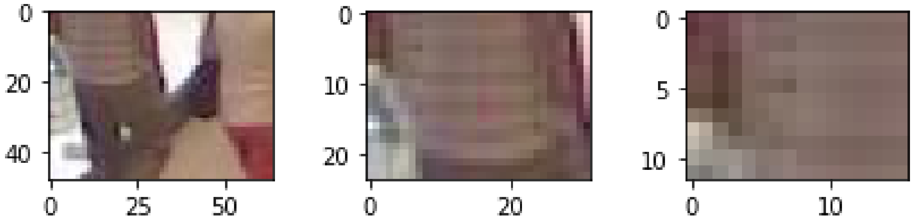


**Fig. 7.** Segments after splitting an original image on 100, 400 and 1600 fragments

### 3.3. Perspective Map

In Mall Dataset the camera that took the pictures was not positioned perpendicularly to the scene, therefore objects on the pictures may have significant distortion. Mall Dataset provides this information in the form of perspective map. The perspective map indicates the real size of each pixel of the images. Figure 8 shows a sample image from the dataset whose fragments are adjusted according to the perspective map.

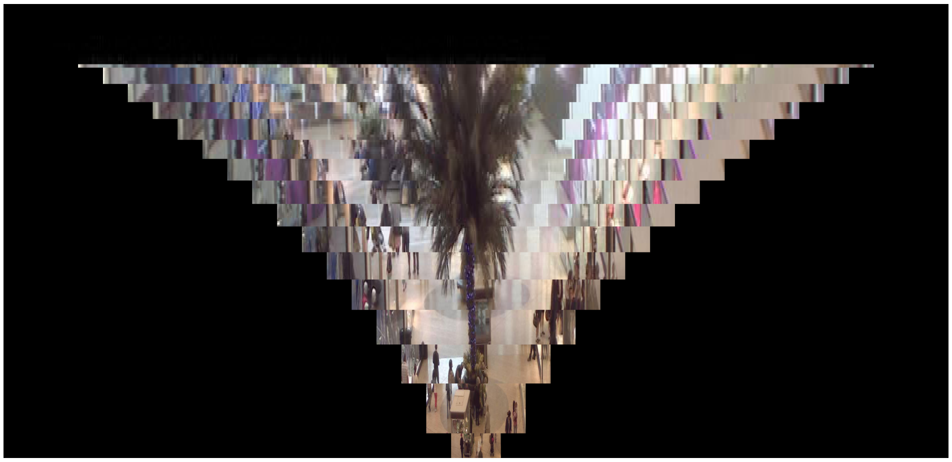


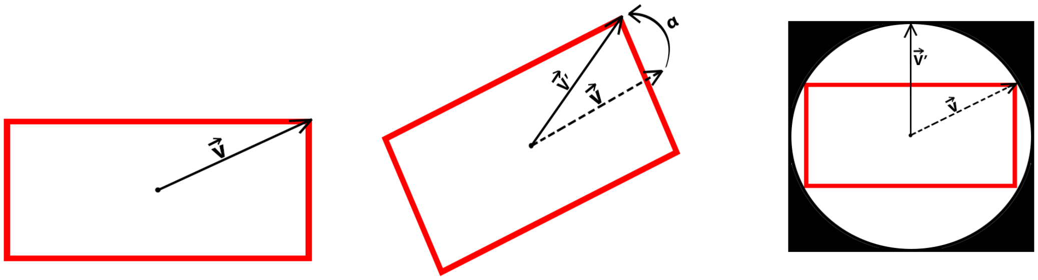
**Fig. 8.** An image with adjusted perspective using perspective map

### 3.4. Data Augmentation

Due to the fact that the number of segments with a person is much smaller than the number of segments without a person (the ratio is equal to $0.0679$), the two classes are unbalanced and a classifier may favor the larger class. Therefore, some augmentation techniques had to be used to produce more balanced classes.

The segments were transformed using random rotations of images by an angle smaller or equal to $\frac{\pi}{12}$. The problem with the standard method of data augmentation is, that after a rotation the segment is cropped to the shape of (a non-rotated) rectangle, whereas new fragments which occur after the rotation are filled out with meaningless values. To prevent this and to create completely augmented segments, a larger surrounding fragment was rotated. Thanks to this approach we did not lose any data after rotation. Figure 9 shows how a rotation of a segment should be performed to not lose information from the surrounding image.

There are also segments for augmentation that lie on the edges of an image — they were augmented in the same way, except that areas of the square which were out of the image were filled with zeros.



(a) Vector $v$ (b) Vector $v$ after rotation by $\alpha$ (c) A new image segment

**Fig. 9.** A segment within the surrounding image

### 3.5. Hardware

We trained our models on an Intel Xeon workstation with 256GB RAM and equipped with four NVIDIA Tesla K80 units (a single K80 unit consists of two interconnected K40 units).

### 3.6. Error Metrics

In order to compare the models, we use two fundamental error metrics — Mean Square Error (MSE) (30), and Mean Absolute Error (MAE) (31) (see for example: [31]), which are calculated according to the following formulas:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 \tag{30}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - x_i| \tag{31}$$

Where $\langle x_1, x_2, \ldots, x_n\rangle$ are the predicted values, whereas $\langle y_1, y_2, \ldots, y_n\rangle$ are the reference (i.e. true) values. In addition, we calculate the accuracy of a prediction $x_i$ with Equation (32)

$$acc(x_i, y_i) = \begin{cases} 1 - \frac{|x_i - y_i|}{y_i} & \text{if } x \in \langle 0, 2y_i\rangle \\ 0 & \text{if } x \in (2y_i, +\infty) \end{cases} \tag{32}$$

and the accuracy of a model as the average over all its predictions.

## 4. Models

As mentioned earlier we focused on building possibly simple models that can be run on low computational power/memory devices, without sacrificing much on the accuracy. Two major limiting factors for our architectures were: a) we could not afford working with full images, therefore the images had to be split into smaller segments, b) in deep neural network architectures, we could not afford any variant of skipping connections (because they force the device to cache the data along the forward-pass of the network). We experimented with variety of architectures of neural networks to predict if a single segment of an image contains a person. For the results presented at the conference We trained 17 different models in total. Two best models are presented in Subsection 4.1 and Subsection 4.2 (see Figure 10). The first one operates on raw (segments) of images, whereas the second makes use of perspective map. Other models mostly differ in the values of regularisation coefficients and the number of segments each image was splitted[4]. To get the final prediction of the size of crowd, we used a cut-off point of each prediction, and then sum over all values of segments that constitute the image.

The model described in Subsection 4.3 enhances the above approach in the following way. Instead of the binary cut-off point, we use a cut-off point based on continuous logic: a value above the cut-off point is truncated to 1, whereas values below the cut-off point are scaled linearly. We trained 12 models in total according to this approach. The best model is described in Subsection 4.3 (see Figure 11). By using this approach we were able to greatly simplify the overall structure of the model and improve its accuracy.

### 4.1. Convolutional Model v1

The architecture of Convolutional model v1 is presented on Figure 10(a). The building blocks of the model are as follows: 2-dimensional convolutional layers [26] [16] with

[4] We also tested a split into 1600 segments, which gave much worse results

receptive field size of $3 \times 3$, stride of $1 \times 1$, $(2, 2)$-max pooling layers [12] [24], Batch normalization layers [15], Dropout layers [27] [3] with parameter $0.5$ and dense (i.e. fully connected) layers. Next to the name of the layers, the numbers specify the size of the input and the size of the output respectively (width, height and depth). We used segmentation technique as described in Subsection 3.2 (each image was split into $400$ segments) and data augmentation as described in Subsection 3.4 to balance the classes of positive and negative appearances.

Because we used a convolutional layer as the first layer in the network, the system had to store segments as full tensors rather than mere vectors. In the result the underlying filters had to be represented as full tensors as well and used in convolution process. Such situation resulted in high memory consumption, and we were forced switch in the training process from batches to mini-batches. The neurons from the hidden layers used Rectified Linear Unit as the activation function, whereas the output layer used the sigmoid function. We used a batch normalization behind each layer which allowed us to set higher values of learning rate at the beginning of neural network learning process, and also prevented Internal Covariate Shift process (the issue is discussed in [15]). The training process utilises two regularisation techniques: dropout connections as described in the above, and l2 regularisation. We tested several regularisation coefficients for the learning process, but found that the value of $10^{-7}$ performs best.

We trained the model with Adam optimiser initially setting the learning rate to $10^{-1}$ to speed up training process. High value of learning rate was acceptable due to Batch Normalization usage. We trained the model for about 75 epochs till accuracy for validation set started getting worse. Then we applied a learning rate value $10^{-3}$ and continued training for next 30 epochs. At that point we exceeded previous results. We decided to settle much lower learning rate - $10^{-5}$ and continue the process for next 20 epochs. At that point we reached the best weights. Further training caused overfitting.

The cut-off point has been chosen to $0.3$. The overall accuracy of the model is presented in Table 3. We achieved Mean Absolute Error (MAE) of $3.44$ and Mean Squared Error (MSE) of $18.87$.

### 4.2. Convolutional Model v2

Convolutional model v2 (see Figure 10(b)) is an enhanced version of the model from the previous subsection. Together with a 3-channel image it additionally utilises the fourth channel with the perspective map of the image. Because the size of the input has significantly grown, a deeper model than the previous one performed better. All of the parameters of the architecture remained unchanged. Data augmentation had to be modified to incorporate the perspective map.

The training process of convolutional model v2 was similar to the training process of convolutional model v1. At the beginning we settled learning rate at value $10^{-3}$ and started training for 30 epochs. After that, we lowered learning rate to value $10^{-4}$ to move slowly to local minimum and continued for the next 20 epochs. Then we lowered learning rate to the value $10^{-5}$ and continued for the next 80 epochs. At that point we reached local minimum and further training had not give any better results.

We experimented with several l2 regularisation hyperparameters for this model as well. Value $10^{-6}$ performed best for this model.

The cut-off point has been chosen to $0.6$. The overall accuracy of the model is presented in Table 2. We achieved Mean Absolute Error (MAE) of $3.35$ and Mean Squared Error (MSE) of $18.33$, which is slightly better than with the previous model.

### 4.3. Continuous convolutional model

Continuous convolutional model (see Figure 11) is a new model based on enhanced cut-off point approach. Just like Convolutional model v2, together with a 3-channel image it additionally utilises the fourth channel with the perspective map of the image. Because the model can use a continuous spectrum of information, we could simplify its architecture by reducing the convolutional blocks to a single layer and remove two dense layers.

The training process of continuous convolutional model was similar to the training process of convolutional model v1 and v2. We started with learning rate at value $10^{-4}$ and trained model for $5$ epochs. After that, we lowered learning rate to value $10^{-5}$ and continued for the next $50$ epochs. Then we again lowered learning rate to the value $10^{-6}$ and continued for the next $20$ epochs. Then we switched back to $10^{-5}$ learning rate value and continued for the next $5$ epochs. Then we settled learning rate at value $10^{-6}$ and continued training for the next $11$ epochs. During last $4$ epochs of the training process model was oscillating close to some local minimum, so we decided to finish the training process.

Several l2 regularisation hyperparameters were tested. Value $10^{-7}$ performed best for this model.

The continuous cut-off point has been chosen to $0.9$. The overall accuracy of the model is presented in Table 2. We achieved Mean Absolute Error (MAE) of $2.46$ and Mean Squared Error (MSE) of $9.6$. These values are far better than the best values achieved by our previous models.

## 5. Conclusions and Future Work

The accuracy and the errors of each of our models are summarised in Table 2. Convolutional model v2 slightly outperforms Convolutional model v1 both in terms of accuracy, MSE and MAE at the cost of a slight increase of model complexity. Nonetheless, it is the continuous convolutional model that beats Convolutional model v1 and v2 in terms of accuracy, MSE, MAE and model simplicity. Table 3 compares our best model to seven most promising models found in literature: the state-of-the art solution by Walach and Wolf [29], the model based on cumulative attributes [10], the model of random projection

**Table 2.** Accuracy and error of our models

| Model | Accuracy | MSE | MAE |
|---|---|---|---|
| Convolutional model v1 | 88.73 | 18.87 | 3.44 |
| Convolutional model v2 | 88.90 | 18.33 | 3.35 |
| Continuous convolutional model | 92.00 | 9.61 | 2.46 |

**Table 3.** Model comparison

| Model | MSE | MAE |
|---|---|---|
| Convolutional model v1 | 18.87 | 3.44 |
| Convolutional model v2 | 18.33 | 3.35 |
| Continuous convolutional model | 9.61 | 2.46 |
| State-of-the-art [29] | NA | 2.01 |
| Cumulative Attributes [10] | 17.7 | 3.43 |
| Count forest [20] | 10.0 | 2.50 |
| Random projection forest [32] | 15.5 | 3.22 |
| Mixture of Counting CNNs [17] | 13.4 | 2.75 |
| Weighted VLAD [25] | 13.05 | 2.86 |
| Localized crowd counting [11] | 15.7 | 3.15 |

forest [32], mixture of counting CNNs [17], weighted VLAD [25] and localized crowd counting [11].

As it turns out, all of our models show competitive results, despite their simplicity and low-computational-cost of their architecture. Our continuous convolutional model is second-best falling behind the state-of-the art solution only. Although our MAE of 2.46 is worse than MAE of 2.01 (the state-of-the art solution), this is negligible from the perspective of crowd-sizes of about 50 persons (i.e. the difference between the errors is less than 1%). Therefore, it may be successfully run on devices that have relatively small resources available. Moreover, we did not use any ensemble methods to tune up our models. By substituting the usual cut-off point by continuous cut-off point we were able to outperform not only our previous models: convolutional model v1 and v2, but also all of the other models evaluated on MALL dataset.

More optimisation techniques may be applied to our models to obtain even lighter architectures (e.g. reduce the size of the color channels, MobileNet architectures [13, 8]). We leave this for the future work. From the perspective of crowd management another interesting direction of research is to identify more details about crowd beside the mere number of the participants. The theory developed by Pierre Bourdieu in [6] tells us that the aspects like gender, race or social class of the individuals may play crucial role in crowd management. We also leave this for future work.

## References

1. Mall dataset. http://personal.ie.cuhk.edu.hk/˜cloy/downloads_mall_dataset.html (2014)
2. Van der Aalst, W.M., Rubin, V., Verbeek, H., van Dongen, B.F., Kindler, E., Günther, C.W.: Process mining: a two-step approach to balance between underfitting and overfitting. Software & Systems Modeling 9(1), 87 (2010)
3. Ba, J., Frey, B.: Adaptive dropout for training deep neural networks. In: Advances in Neural Information Processing Systems. pp. 3084–3092 (2013)
4. Bonyadi, M.R., Michalewicz, Z., Przybylek, M.R., Wierzbicki, A.: Socially inspired algorithms for the travelling thief problem. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation. pp. 421–428. ACM (2014)
5. Boominathan, L., Kruthiventi, S.S., Babu, R.V.: Crowdnet: A deep convolutional network for dense crowd counting. In: Proceedings of the 24th ACM international conference on Multimedia. pp. 640–644. ACM (2016)

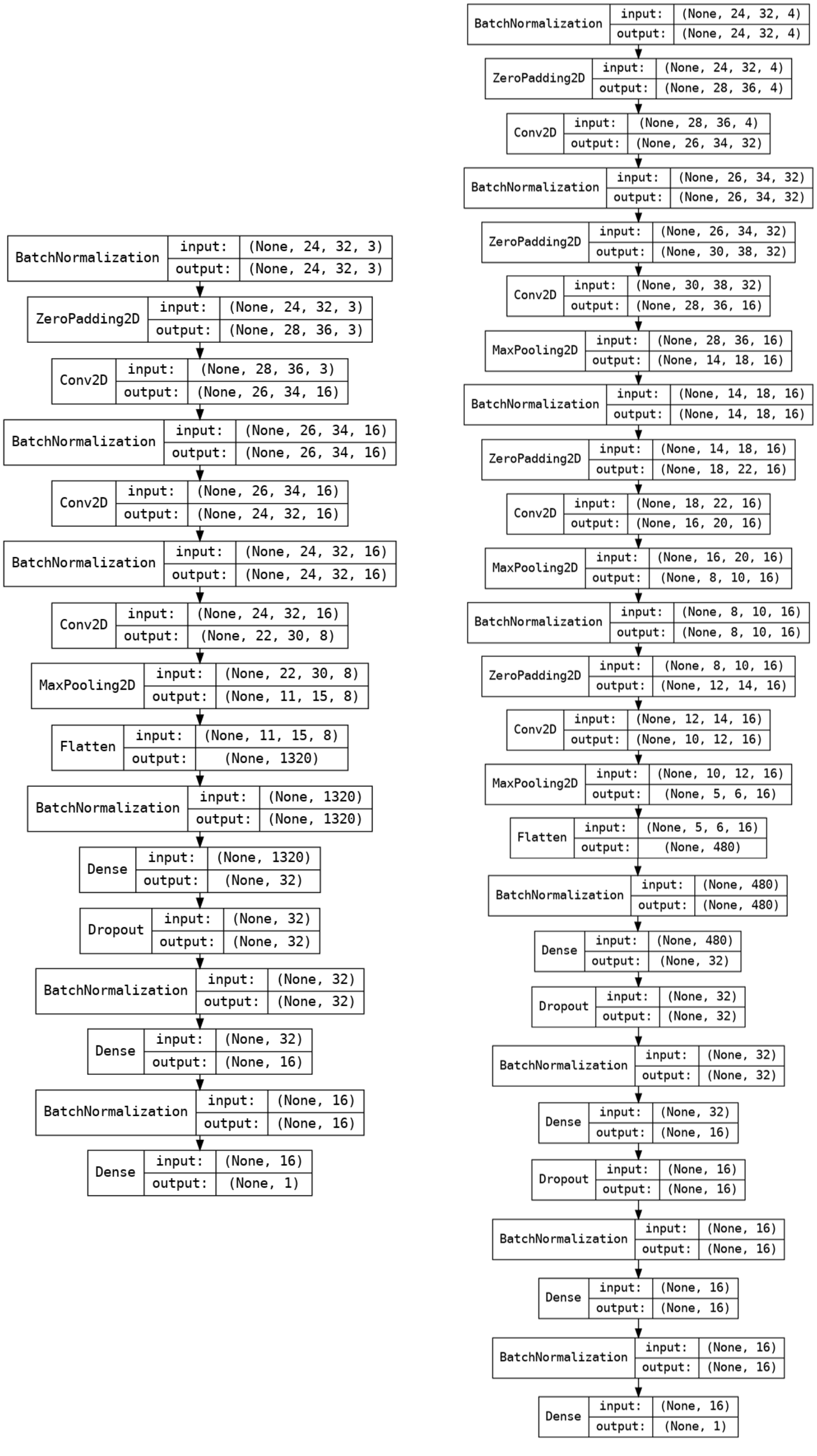
6. Bourdieu, P.: Distinction: A social critique of the judgement of taste. Routledge (2013)
7. Brzeski, A., Grinholc, K., Nowodworski, K., Przybylek, A.: Evaluating performance and accuracy improvements for attention-ocr. In: 18th International Conference on Computer Information Systems and Industrial Management Applications (2019)
8. Brzeski, A., Grinholc, K., Nowodworski, K., Przybylek, A.: Residual mobilenets. In: Workshop on Modern Approaches in Data Engineering and Information System Design at ADBIS (2019)
9. Change Loy, C., Gong, S., Xiang, T.: From semi-supervised to transfer counting of crowds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2256–2263 (2013)
10. Chen, K., Gong, S., Xiang, T., Change Loy, C.: Cumulative attribute space for age and crowd density estimation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2467–2474 (2013)
11. Chen, K., Loy, C.C., Gong, S., Xiang, T.: Feature mining for localised crowd counting. In: BMVC. vol. 1, p. 3 (2012)
12. Graham, B.: Fractional max-pooling. arXiv preprint arXiv:1412.6071 (2014)
13. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
14. Idrees, H., Saleemi, I., Seibert, C., Shah, M.: Multi-source multi-scale counting in extremely dense crowd images. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2547–2554 (2013)
15. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105 (2012)
17. Kumagai, S., Hotta, K., Kurita, T.: Mixture of counting cnns: Adaptive integration of cnns specialized to specific appearance for crowd counting. arXiv preprint arXiv:1703.09393 (2017)
18. Leibe, B., Seemann, E., Schiele, B.: Pedestrian detection in crowded scenes. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol. 1, pp. 878–885. IEEE (2005)
19. Loy, C.C., Chen, K., Gong, S., Xiang, T.: Crowd counting and profiling: Methodology and evaluation. In: Modeling, simulation and visual analysis of crowds, pp. 347–382. Springer (2013)
20. Pham, V.Q., Kozakaya, T., Yamaguchi, O., Okada, R.: Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3253–3261 (2015)
21. Przybylek, K., Shkroba, I.: Crowd counting á la bourdieu. In: European Conference on Advances in Databases and Information Systems. pp. 295–305. Springer (2019)
22. Przybylek, M.R., Wierzbicki, A., Michalewicz, Z.: Decomposition algorithms for a multi-hard problem. Evolutionary computation 26(3), 507–533 (2018)
23. Przybylek, M.R., Wierzbicki, A., Michalewicz, Z.: Multi-hard problems in uncertain environment. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016. pp. 381–388. ACM (2016)
24. Scherer, D., Müller, A., Behnke, S.: Evaluation of pooling operations in convolutional architectures for object recognition. In: International conference on artificial neural networks. pp. 92–101. Springer (2010)
25. Sheng, B., Shen, C., Lin, G., Li, J., Yang, W., Sun, C.: Crowd counting via weighted vlad on a dense attribute feature map. IEEE Transactions on Circuits and Systems for Video Technology 28(8), 1788–1797 (2016)
26. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
27. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1), 1929–1958 (2014)

28. Stewart, R., Andriluka, M., Ng, A.Y.: End-to-end people detection in crowded scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2325–2333 (2016)
29. Walach, E., Wolf, L.: Learning to count with cnn boosting. In: European Conference on Computer Vision. pp. 660–676. Springer (2016)
30. Wang, M., Li, W., Wang, X.: Transferring a generic pedestrian detector towards specific scenes. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3274–3281. IEEE (2012)
31. Willmott, C.J., Ackleson, S.G., Davis, R.E., Feddema, J.J., Klink, K.M., Legates, D.R., O'donnell, J., Rowe, C.M.: Statistics for the evaluation and comparison of models. Journal of Geophysical Research: Oceans 90(C5), 8995–9005 (1985)
32. Xu, B., Qiu, G.: Crowd density estimation based on rich features and random projection forest. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 1–8. IEEE (2016)

**Karolina Przybyłek** obtained her master's degree in social sciences (with honors) from University of Warsaw in 2018. Since then she has been dealing with scientific and social activities: she co-organized and participated in international scientific conferences. She is currently preparing her PhD dissertation on the decision-making process of judges. Her research interests include: theories of education, the sociology of Pierre Bourdieu, sociology of art, criminal law, criminal trial, issues connected with the status of courts and judges.

**Illia Shkroba** is a teaching assistant at Polish-Japanese Academy of Information Technology in Warsaw. He obtained his master's degree in Information Technology with Data Science specialization (with honors) from Polish-Japanese Academy of Information Technology in 2020. Since 2019 he is actively participating in government research projects. His research interests include: deep neural networks, data mining, process mining.

*Received: January 15, 2020; Accepted: September 1, 2020.*


BatchNormalization input: (None, 24, 32, 3) output: (None, 24, 32, 3)
ZeroPadding2D input: (None, 24, 32, 3) output: (None, 28, 36, 3)
Conv2D input: (None, 28, 36, 3) output: (None, 26, 34, 16)
BatchNormalization input: (None, 26, 34, 16) output: (None, 26, 34, 16)
Conv2D input: (None, 26, 34, 16) output: (None, 24, 32, 16)
BatchNormalization input: (None, 24, 32, 16) output: (None, 24, 32, 16)
Conv2D input: (None, 24, 32, 16) output: (None, 22, 30, 8)
MaxPooling2D input: (None, 22, 30, 8) output: (None, 11, 15, 8)
Flatten input: (None, 11, 15, 8) output: (None, 1320)
BatchNormalization input: (None, 1320) output: (None, 1320)
Dense input: (None, 1320) output: (None, 32)
Dropout input: (None, 32) output: (None, 32)
BatchNormalization input: (None, 32) output: (None, 32)
Dense input: (None, 32) output: (None, 16)
BatchNormalization input: (None, 16) output: (None, 16)
Dense input: (None, 16) output: (None, 1)
BatchNormalization input: (None, 24, 32, 4) output: (None, 24, 32, 4)
ZeroPadding2D input: (None, 24, 32, 4) output: (None, 28, 36, 4)
Conv2D input: (None, 28, 36, 4) output: (None, 26, 34, 32)
BatchNormalization input: (None, 26, 34, 32) output: (None, 26, 34, 32)
ZeroPadding2D input: (None, 26, 34, 32) output: (None, 30, 38, 32)
Conv2D input: (None, 30, 38, 32) output: (None, 28, 36, 16)
MaxPooling2D input: (None, 28, 36, 16) output: (None, 14, 18, 16)
BatchNormalization input: (None, 14, 18, 16) output: (None, 14, 18, 16)
ZeroPadding2D input: (None, 14, 18, 16) output: (None, 18, 22, 16)
Conv2D input: (None, 18, 22, 16) output: (None, 16, 20, 16)
MaxPooling2D input: (None, 16, 20, 16) output: (None, 8, 10, 16)
BatchNormalization input: (None, 8, 10, 16) output: (None, 8, 10, 16)
ZeroPadding2D input: (None, 8, 10, 16) output: (None, 12, 14, 16)
Conv2D input: (None, 12, 14, 16) output: (None, 10, 12, 16)
MaxPooling2D input: (None, 10, 12, 16) output: (None, 5, 6, 16)
Flatten input: (None, 5, 6, 16) output: (None, 480)
BatchNormalization input: (None, 480) output: (None, 480)
Dense input: (None, 480) output: (None, 32)
Dropout input: (None, 32) output: (None, 32)
BatchNormalization input: (None, 32) output: (None, 32)
Dense input: (None, 32) output: (None, 16)
Dropout input: (None, 16) output: (None, 16)
BatchNormalization input: (None, 16) output: (None, 16)
Dense input: (None, 16) output: (None, 16)
BatchNormalization input: (None, 16) output: (None, 16)
Dense input: (None, 16) output: (None, 1)

(a) Convolutional model v1

(b) Convolutional model v2

**Fig. 10.** CNN models

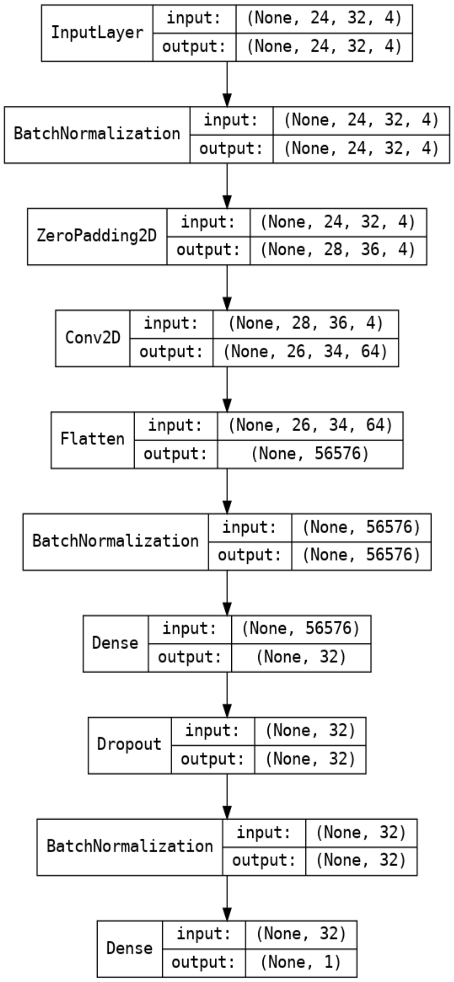


**Fig. 11.** Continuous convolutional model

Computer Science and Information Systems 17(3):983–1006 https://doi.org/10.2298/CSIS200115030K

# Visual E-Commerce Values Filtering Framework with Spatial Database metric [⋆]

M. Kopecky and P. Vojtas[⋆⋆]

Dpt. of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague
{kopecky|vojtas}@ksi.mff.cuni.cz

**Abstract.** Our customer preference model is based on aggregation of partly linear relaxations of value filters often used in e-commerce applications. Relaxation is motivated by the Analytic Hierarchy Processing method and combining fuzzy information in web accessible databases. In low dimensions our method is well suited also for data visualization.
The process of translating models (user behavior) to programs (learned recommendation) is formalized by Challenge-Response Framework ChRF. ChRF resembles remote process call and reduction in combinatorial search problems. In our case, the model is automatically translated to a program using spatial database features. This enables us to define new metrics with visual motivation.
We extend the conference paper with inductive ChRF, new representation of user and an additional method and metric. We provide experiments with synthetic data (items) and users.

**Keywords:** E-commerce values filtering, spatial database, recommender systems, user preference learning, experiments, synthetic data, spatial evaluation measures

## 1. Introduction, motivation, contributions

Our main motivation are recommender systems so far they point us to interesting items on e-commerce sites. Such a system has to be personalized to each user/customer preferences separately. We are modeling user by its behavior (rating) on visited items and expect (inductive) programs to be able to generalize this behavior to all items. We measure success of this generalization by several spatial database metrics.

In representation of customer preferences we restrict to Fagin-Lotem-Naor-class of models (FLN models). R. Fagin, A. Lotem and M. Naor in their paper [8] described a (middleware) top-k query system where each object in a database has $m$ scores, one for each of $m$ attributes (somewhere out in the web) that represent relevance degrees. To each object is then (on the middleware) assigned an overall score that is obtained by combining the attribute scores using a fixed monotone combining rule. This approach enables multicriterial ordering.

[⋆] This is an extended version of a conference paper [13] Kopecky M., Vojtas P. (2019) Graphical E-Commerce Values Filtering Model in Spatial Database Framework. In: Welzer T. et al. (eds) New Trends in Databases and Information Systems. ADBIS 2019. Communications in Computer and Information Science, vol 1064. Springer, Cham. pp 210-220

[⋆⋆] Supported by Czech grant Progres Q48, We work on idea to use these types of models for e-commerce value filtering. Our goal is to present intuitiveness of visual features of these models and automated translation to programs. It is also suitable for implementation of ”best match” in case, when system has to respond ”we were not able to find any matching results, but we found these similar listings for you” (see e.g. [15]). This motivated softening, relaxing value filtering. Original motivation for [8] was multimedia search where attributes are inherently fuzzy (hence relaxed). Motivation comes also from combination of fuzzy information as developed along the IBM Almaden project Garlic ([7]) and top-k querying of web accessible databases ([15]). Degree of relaxation is motivated also by the AHP - Analytic Hierarchy Processing method ( [20]). In AHP relaxation is often done by a domain expert - here we need fast automatic response.

The process of translating models to programs is formalized by Challenge-Response Framework ChRF. ChRF resembles many problem reduction scenario.

To have our models intuitive, we visualize them - the price we have to pay is we can depict only two or three dimensions. The idea is to use most important attributes and/or some aggregated ones. In our case, the model is automatically translated to a program using spatial database features.This enables us to define new metrics with spatial motivation.

We provide offline experiments with synthetic data (items) and users. This is an extended version of our conference paper [13]. This paper was extended with additional model, method, metric, experiments and lot of new comparisons.

Main contributions of this paper are those of the original conference paper and new ones ( + sign denotes additional extensions):

- Spatial representation of linear Fagin-Lotem-Naor model for most important attributes
- Challenge-Response Framework ChRF for translating models to programs - original conference paper version + is extended with inductive ChRF
- Visual aspects of our model enabling e-commerce customer user studies
- Spatial metrics - area based + new item size based
- Spatial methods of user preference learning - pivot based + new
- Representation of user using synthetic data or additionally a new representation using convex hull of most preferred objects
- Prototype and four types of experiments - two methods versus two metrics over multiple users
- Experiments with synthetic data (items) and users

Paper is organized as follows: Section 2. deals with visual, linear, multiuser, content based Fagin-Lotem-Naor class of models. Section 3. describes basic Challenge Response Framework ChRF and its inductive version. Section 4. is on data, methods, spatial metric and experiments. Here we describe practical challenge-response construction for our experiments. Detailed description of spatial SQL computing our metrics is also provided. We briefly mention related research and add conclusions and future work.

## 2. Visual, linear, multiuser, content based FLN models

First we describe Fagin-Lotem-Naor class of models. In [8] they assume, that each object $o$ has assigned $m$-many attribute scores $x_i^o \in [0; 1]$. A typical example is that this score is coding order of access when querying multiple web-accessible databases. Combination function $t : [0; 1]^m \longrightarrow [0; 1]$ is assumed to fulfill: $t(0, \ldots, 0) = 0$, $t(1, \ldots, 1) = 1$ and $t$ preserves ordering, i.e. if $x_j \leq_{[0,1]} y_j$ for all $1 \leq j \leq m$ then

$$t(x_1, \ldots, x_j, \ldots, x_m) \leq_{[0,1]} t(y_1, \ldots, y_j, \ldots, y_m) \tag{1}$$

Because of this inequality we call this function monotone. The overall score of object $o$ is $t(o) = t(x_1^o, \ldots, x_j^o, \ldots, x_m^o)$.

In [14] we have described a class of LT-linear triangular models which is a subclass of FLN models. Especially, such a model can be generated by domain preference functions $f_i : D_i \longrightarrow [0, 1]$ and $x_i^o = f_i(o.A_i)$ and a combination function $t$. To be able to process such models by a spatial database it is suitable to have these functions linear or at least partially linear. Special case of such domain preference function are triangular (or trapezoidal) functions which can be considered as softening / relaxing of value filters in e-commerce.

In Figure 1 there are two such preference algorithms - the green one (user $u$) $\alpha$ (given by $f_1, f_2$ and $c_{2/3}^u = (t^u)^{-1}(2/3)$) and the red algorithm (user $v$) $\beta$ (given by $g_1, g_2$ and $c_{2/3}^v = (t^v)^{-1}(2/3)$). We can think of $\alpha$ as being the correct model and of $\beta$ as being the computed (learned) model. This should illustrate a decision maker (e-shop owner, customer) situation when comparing two decision alternatives. In this figure, attribute preferences are triangular - this is a softening of one element value filter. There are two aggregations $\frac{2*x_1+x_2}{3}$ and $\frac{2*x_2+x_1}{3}$ represented by $2/3$ contour line in the preference cube.

Main feature of our model is visualization of these contour lines in the data cube. Each contour line corresponds to the polygon in the data cube. Using attribute preferences can be endpoints of preference cube contour lines traced back (by respective horizontal and vertical lines and their intersections with attribute preferences) to respective attribute values in domains (in our figure there are always two of them). This should be intuitive for user / customer. Here we see areas in data cube (with preference bigger or equal to $2/3$) and area of their intersection.

Another aspect of illustration of Figure 1: - the green one (user $u$) $\alpha$ (given by $f_1, f_2$ and $(t^u)^{-1}(2/3)$) describes deduction and we can calculate polygon in data cube having preference at least 2/3.

$$P_{\alpha,2/3}^u = \left[t^u(f_1^u(o.A_1), f_2^u(o.A_2))\right]^{-1} (\geq 2/3) \tag{2}$$

Similarly to polygon for the red algorithm (user $v$) $\beta$ describes the inductive procedure - we have only few (explicit ratings) of some visited objects (here $A$ is rated $0.3$ ... $F$ is rated $0.7$) and we try to specify either $g_1^v, g_2^v$ and $(t^v)^{-1}(\geq 2/3)$ or directly polygon $P_{\beta,2/3}^v$. Instead of 2/3 we can consider arbitrary level of preference $h \in [0, 1]$.

Let us stress here that this inductive task can not be considered as high dimensional regression, because we need ordering on each server separately (in typical situation of web accessible databases ([15])).

We will consider three metrics motivated by this spatial representation of relaxed value filters - the area metrics, number of data points metric and metric calculating with average distribution of produced items given by a measure. Having the ”correct” model $P_h^u$ given by data and computed (learned) $\hat{P}_{\alpha,h}^u$, it is natural to ask for precision and recall of such models at different levels.

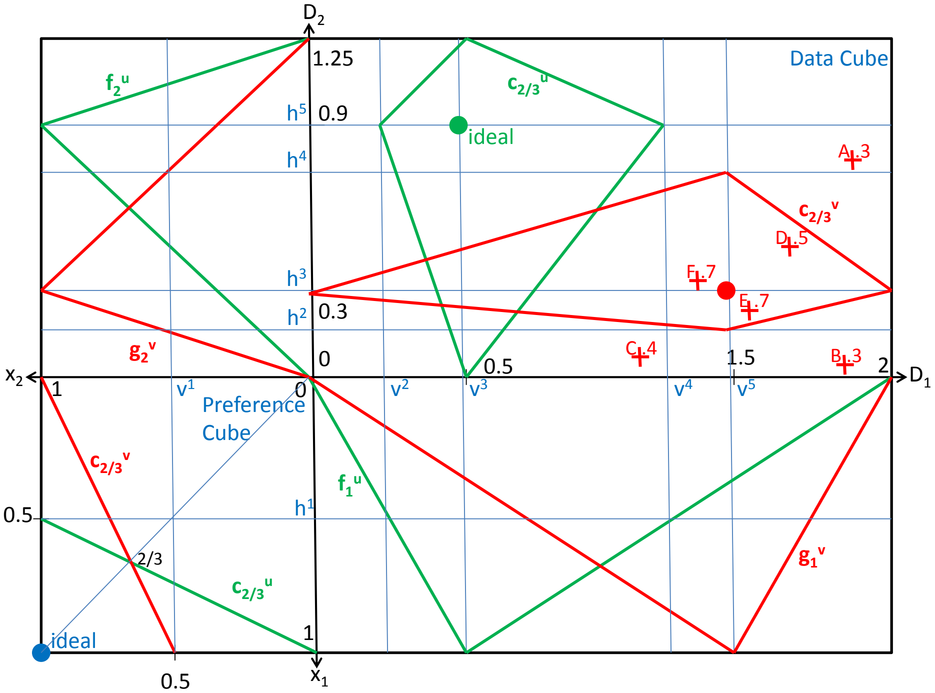


**Fig. 1.** We illustrate both deductive (in green) and inductive (in red) aspects of our linear FLN-class model. More description in the text.

We will provide experiments with two metrics, area based metric and item size based metric. Area precision and recall of algorithm $\alpha$ for user $u$ at preference level $h$ are expressed as

$$AP^u_{\alpha,h} = \frac{area(P^u_h \cap \hat{P}^u_{\alpha,h})}{area(\hat{P}^u_{\alpha,h})} \qquad (3) \qquad\qquad AR^u_{\alpha,h} = \frac{area(P^u_h \cap \hat{P}^u_{\alpha,h})}{area(P^u_h)} \qquad (4)$$

When having some items data $D$, instead of area metrics we can calculate these fractions by number of data points in each area and we get item size based precision and recall.

$$IP^u_{\alpha,h} = \frac{|D \cap P^u_h \cap \hat{P}^u_{\alpha,h}|}{|D \cap \hat{P}^u_{\alpha,h}|} \qquad (5) \qquad\qquad IR^u_{\alpha,h} = \frac{|D \cap P^u_h \cap \hat{P}^u_{\alpha,h}|}{|D \cap P^u_h|} \qquad (6)$$

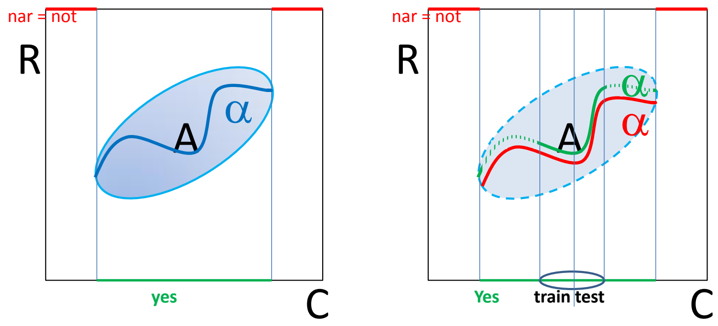


**Fig. 2.** Illustration of basic (left) and inductive (right) ChRF situation

Sometimes we know the measure $\mu$ of distribution of production of items (data points). In this case instead of area or number of data points we can use distribution based precision and recall.

$$DP^u_{\alpha,h} = \frac{\int_{P^u_h \cap \hat{P}^u_{\alpha,h}} x d\mu}{\int_{\hat{P}^u_{\alpha,h}} x d\mu} \quad (7) \qquad DR^u_{\alpha,h} = \frac{\int_{P^u_h \cap \hat{P}^u_{\alpha,h}} x d\mu}{\int_{P^u_h} x d\mu} \quad (8)$$

## 3. ChRF - Challenge Response Framework

In this chapter we will deal with a procedure known from many environments - as problem reduction, compilation based approach, many-one reduction, client server RPC, requester-helper in agent systems, question-answer - in all of these reduction converts instances of one decision/search problem into instances of a second decision/search problem. The solution of this second problem is then transformed to a solution of the starting one. We will adopt a general term for these - challenge-response framework.

In this chapter we will describe the Challenge Response Framework which will serve as a formal tool for our future activities. First we describe the case when the requester does not know anything about solution and takes whatever the solver answers. Second we will consider the case when we have already some partial information (usually examples data in supervised learning). This can help to choose the best helper/solver.

Specific form of Challenge Response Framework suitable for our multi-user data and spatial data metric will be introduced in experiments chapter.

### 3.1. Basic Challenge Response Framework

Motivated by an old mathematical idea from [21] we use the terminology arousing from [4] and define the Challenge Response Framework ChRF. Our goal is to use ChRF as a formal framework for description of translation of models to programs.

Challenge Response Situation $S = (C, R, A)$ consists of a set $C$ of challenges, set of responses $R$ and an acceptability relation $A \subseteq C \times R$ (can be preferential). For an

$c \in C, r \in R$ we read $A(c,r)$ as "$r$ is an acceptable response (in some degree) for challenge $c$". We assume, that each set $R$ contains also a special element $nar$ representing "there is no acceptable response".

We assume: $A(c, nar)$ is equivalent to $(\forall r \in R \backslash \{nar\})(\neg A(c,r))$. The set $R \backslash \{nar\}$ are meaningful responses, $nar$ is like logical "not" in combinatorial decision problems. See, Fig. 2 left.

Challenge Response Reduction of a situation $S_1 = (C_1, R_1, A_1)$ to a situation $S_2 = (C_2, R_2, A_2)$ consists of a pair of functions $(f^-, f^+)$ such that $f^- : C_1 \longrightarrow C_2$, $f^+ : R_2 \longrightarrow R_1$, such that $f^+(nar_2) = nar_1$, $f^+(r_2) = nar_1$ implies $r_2 = nar_2$ and following holds:

$$(\forall c_1 \in C_1)(\forall r_2 \in R_2)(A_2(f^-(c_1), r_2) \longrightarrow A_1(c_1, f^+(r_2)) \qquad (9)$$

see Figure 3. Let us stress that we require here only implication. In decision problems we need equivalence, in search problems we have to prevent from fake reduction (when the implication can be true even when there is no acceptable response). This is the main reason we have introduced the "nar" element and we require that 9 holds also for "nars".

Note that $A_2(f^-(c_1), nar_2) \longrightarrow A_1(c_1, nar_1)$ is equivalent to $\neg A_1(c_1, nar_1) \longrightarrow \neg A_2(f^-(c_1), nar_2)$ and this to $(\forall c_1 \in C_1)(\exists r_1 \in R_1 \setminus \{nar_1\})(A_1(c_1, r_1)) \longrightarrow (\exists r_2 \in R_2 \setminus \{nar_2\})(A_2(f^-(c_1), r_2))$. See, Fig. 3

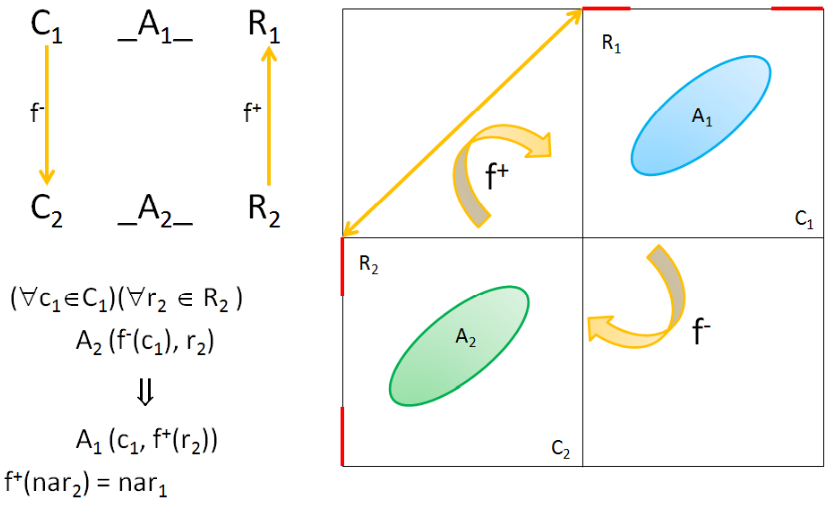


**Fig. 3.** Illustration of basic ChRF reduction.

Notice that we can consider two levels of understanding challenges and responses. First is on the level of each situation itself. Each situation has own challenge instances and responses. A level higher we can consider the situation $S_1$ as a sort of challenge (somebody requests help) and the situation $S_2$ provides help (response). This can form a second order structure $\mathcal{S}$ where challenges $C_\mathcal{S}$ are ChR situations, the same responses $R_\mathcal{S}$

$$\begin{tikzcd}
C_{model} \arrow[d, "f^{-}_{m2p}"'] & -A_{model}- & R_{model} \\
C_{program} & -A_{program}- & R_{program} \arrow[u, "f^{+}_{m2p}"']
\end{tikzcd}$$

**Fig. 4.** Model to program transformation as ChRF reduction.

are ChR situations and acceptability relation $A_S$ consists of pair $(S_1, S_2)$ such that $S_1$ is reducible to $S_2$. We are not going deeper into this.

This ChRF reduction can be used to formally represent transformations of models to programs, see Figure 4. Notice that we cannot grasp the reality in full complexity. Here we understand model as a model of reality represented by available data, consisting of various signals, user behavior, etc.

In our model situation challenges can come from user interaction with visual interface (so far not implemented) were the user can slide ideal points, descent of relaxation, combination function etc. Transformation $f^{-}_{m2p}$ has to be fully automated and sends these actions to program situation challenges (inputs). In what follows we describe implementation of $f^{-}_{m2p}$ in Oracle Spatial. Challenges of the program situation are input and responses are output. So far we have implemented two user representations. First calculation of polygons of contour lines from user model in synthetic data set. Second, we represent user by a convex hull of most preferred items (after possible user changes behavior). Respective visualization sends $f^{+}_{m2p}$ back to model response - a possibly user visual interface .

### 3.2. iChRF - inductive Challenge Response Framework

In this section we describe Challenge Response Framework with partial information. Our main motivation is to have a framework for handling supervised learning. We restrict ourselves to regression tasks, where we assume the example set in the form of a function $E(\bar{x}) = y$ with a vector of independent variables $\bar{x}$ and a dependent variable $y$. See Fig. 2 right.

A straightforward usage of ChRF is not fully satisfactory because we need a machinery for calculation error of learning (in contrast to basic ChRF where the truth of implication is understood as in mathematics).

We are not going to describe various aspects of the learning process. Here we assume that we have a class of machine learning algorithms/programs $\Pi$. $\Pi$ can consist e.g. of linear regression, logistic regression, decision tree, SVM, .... Each $\alpha \in \Pi$ has a set of hyperparameters $H^{\alpha}$ and for each $h \in H^{\alpha}$ there is a program $\alpha^{h}$ which is a candidate for generalization of set of examples. This program generates a program situation

$$S_{\alpha^h} = \left(In_{\alpha^h}, Out_{\alpha^h}, \alpha^h\right). \tag{10}$$

iChR reduction will be used to calculate the quality of approximation of the example set. Reduction will reduce the challenge of generalizing the example data set (model) to program situation.

Assume we have an example set $E$, then the model situation looks like

$$S_E = (C_E, R_E, A_E) \tag{11}$$

where $C_E$ is the set of independent variable vectors $\bar{x}$ from the domain of example set $E$. To define response set we need a metric $\rho_i$ for calculating individual error (sometimes it depends on the algorithm $\rho_i^\alpha$ ). $R_E$ is the set of triples (usually of real numbers) $(y, \hat{y}, e))$. The acceptability relation is defined as follows:

$$A_{E,i}(\bar{x}, (y, \hat{y}, e)) \text{ iff } E(\bar{x}) = y \text{ and } e = \rho_i(y, \hat{y}). \tag{12}$$

An Inductive Challenge Response Reduction from $S_E$ to $S_{\alpha^h}$ consists of two identity mappings $f^-, f^+$, where $f^+$ maps estimate $\alpha^h(\bar{x}) = \hat{y}$ to the second coordinate of $R_E$.

The trickier part is calculation of degree of validity of the reduction. The basic ChR reduction is universally quantified over challenges. Here it also makes sense to ask, how good is $p^{\alpha,h}$ in approximating the example set $E$? The truth value of the universally quantified statement $(\forall \bar{x} \in C_E)$ will be calculated by an aggregate measure $\rho_a$ (as usual in data mining, again maybe depending on $\alpha$). So the quality of reduction is measured by

$$\rho_{a,i}^{\alpha,h}(E) = \rho_a\left(\left\{e : \bar{x} \in C_E, \text{ and } A_{E,i}\left(\bar{x}, \left(E\left(\bar{x}\right), \alpha^h(\bar{x}), e\right)\right)\right\}\right). \tag{13}$$

If measures depend on $\alpha$ we write $\rho^{\alpha,h}$.

We can imagine to let run this approximations in parallel (over all algorithms, parameters, Crossvalidation splits and tests) and the winner will be $argmin_{\alpha \in A} argmin_h \rho^{\alpha,h}$....

Example. If $\rho_i(y, \hat{y}) = |y - \hat{y}|$ then
$\rho_{ABS} = \sum_{C_E} |y - \hat{y}|$, $\rho_{AVG} = \frac{\sum_{C_E} |y-\hat{y}|}{|C_m|}$, similarly $\rho_{MAX} = \max_{C_E} |y - \hat{y}|$;
if $\rho_i(y, \hat{y}) = (y - \hat{y})^2$ then $\rho_{RMSE} = \sqrt{\sum \frac{(y-\hat{y})^2}{|C_E|}}$.

In this paper we will consider aggregated error on polygons (contour lines) in data cube.

## 4. Data, methods, spatial metric and experiments

In this section we will describe experiments with two metrics. First area based and the second based on the number of items in respective polygons.

### 4.1. Data

Our experiments are two fold. First are pure synthetic data where we know that data are generated by our model (hence inductive part measures true ability to find the model from global preferences). In the second part we mimic the situation that we have user's behavior, i.e. preferences of visited objects.

Data items are not evenly distributed, but form four distinct clusters near corners of the data cube.

**Fully synthetic data for experiments** In [14] we have studied a subclass of FLN models - Linear combination of Triangular attribute modes - LT-models. LT-models can be generalized to trapezoidal models.

We consider pivot based learning from [14] with some stochastic noise. Results are evaluated through new metrics calculating spatial data characteristics of LT-models.

Our experiments are using user and item data from [14] together with sparse preference matrix $M = \{< 0; 1 > \cup\, null\}^{|U| \times |I|}$ where $U$ is set of users and $I$ is set of items.

Training data for each user $u \in U$ contain only the corresponding row of preference matrix, i.e. set of ranked items with their preferences.

In our simulated environment, each user $u$, is fully represented by the triple $< i_1, i_2, w_1 >$, that can be understood as quadruple
$< i_1, i_2, w_1, w_2 = 1 - w_1 >$, where

- $i_1$ represents the ideal point in first data dimension.
- $i_2$ represents the ideal point in second data dimension.
- $w_1$ represents the weight (importance) of first data dimension.
- $w_2$ represents the weight (importance) of second data dimension.

This way we can know all preferences for all items (although we use only those on visited items, which are also generated randomly).

**Data as from user behavior** Here we use also synthetic user. We calculate behavior on visited objects and this is the only input for rest $S_{model}$ situation. To follow our visualization strategy and to use spatial database metric we calculate the convex hull of these points.

### 4.2. Practical challenge-response construction

Challenges $C_{model}$ in model situation equal to rows of partially filled preference matrix $M$, i.e. known preferences $y$ of each single user $u \in U$ for all visited items $V^u$.

$$C_{model} = \{\{(u, i, y) : i \in V^u\} : u \in U\} \tag{14}$$

or in content based notation item is represented by the vector of attribute values $\bar{x} = \{i.A_1, i.A_2, \ldots, i.A_m\}$. Then set of challenges looks like

$$C_{model} = \{\{(u, \bar{x}, y) : \bar{x} \in V^u\} : u \in U\}\,. \tag{15}$$

$R_{model}$ consists again of triples, first coordinate is the correct value polygon $P^u_h$, the second is the polygon $\hat{P}^u_{\alpha,h}$ computed by $A_{program}$ and third is the measure of acceptance between first and second coordinate (description follows is subsection on metric). First coordinate of $R_{model}$ will differ depending on type of data used for experiments. In case we have synthetic data $R_{model}$ consists of data cube contour lines - polygons $P^u_h$ for chosen levels of preference $h$ (see deductive part of Figure 1). In case we have preference degrees of visited items, first coordinate of $R_{model}$ consists of convex hulls $K^u_h$ of rated data points in data cube with preference at least $h$. Acceptance relation on a preference degree $h$ is defined as follows

$$A_{model,h} = \Big\{ \Big( \{(u,\bar{x},y) : \bar{x} \in V^u\}, (P^u_h, \hat{P}^u_{\alpha,h}, \rho) \Big) : u \in U \Big\} \tag{16}$$

Mappings $f^-_{m2p}$ and $f^+_{m2p}$ are identities on respective domains.

Considering first part of experiments with synthetic data, $\_A_{program}\_$ finds an approximation of the user by choosing closest pivot $m$ in the dataset [14] and computes polygons $\hat{P}^u_p$ from pivots deductive model.

In the second part of experiments, we have only user's preferences on visited objects and $\_A_{program}\_$ finds $\hat{i_1}, \hat{i_2}, \hat{w}_1$ (see inductive part of Figure 1) and computed respective polygon.

Evaluation of implication 9 is here quantified through all users $u \in U$ and done by computing precision and recall on polygons obtained by $A_{model}$ and $A_{program}$.

### 4.3. Methods, programs, algorithms

We provided two different algorithms for modelling user $u$ by an algorithm. First algorithm $M_1$ tries to find closest neighbor pivot from set of equidistantly spread set of pivots $P = \big\{p_k : p_k =< i^k_1, i^k_2, w^k_1 >\big\}$. The second algorithm $M_2$ is based on centre of mass computation.

**Closest neighbor pivots** In this method, further denoted as $M_1$, is each user $u =< i_1, i_2, w_1 >$ with the contour polygons $P^u_h$ estimated by the closest neighbor pivot $p_k =< i^k_1, i^k_2, w^k_1 >$. There exist $11 \times 11 \times 11$ equidistantly distributed pivots that split the data cube to $10 \times 10$ tiles. For each of $11 \times 11$ possible positions there exist 11 pivots with different weights $w_1 \in \{0.0, 0.1, \ldots, 1.0\}$

This method represents supposedly more precise user preference estimation, but its computation is more time-consuming, because it needs to compare known information about each user with each pivot.

To find the closest neighbour pivot, we can use different metrics. Currently, the pivot with minimal average difference in preferences over all ranked items. The chosen pivot defines computed contour polygons $\hat{P}^u_{M_1,h}$.

**Centre of mass** This method, further denoted as $M_2$, induces ideal point of each user according to location of best rated items first. It is much faster to compute, and so it can be dynamically re-computed online during the user activity and adopt immediately any knowledge about user's changing preferences.

In each dimension, known rated items are split to sets of items rated in interval $(0.9; 1.0 >$, $(0.8; 0.9 >$ etc. The interval with at least three items (two or one if not exists) is taken and the average value is computed. We then take this average value as estimated ideal point $\hat{i}_1$, respectively $\hat{i}_2$ in corresponding dimension. Splitting of rated items and taking the highest rated item first gives priority to optimal items, that would probably have smaller variance, and thus allow us to estimate location of ideal point better.

The overall score depends not only on partial preferences in individual dimensions, but also on the weights, that user assigns to individual dimensions – item features. Having estimations of $\hat{i}_1$ and $\hat{i}_2$, for ranked item $o$ we can compute partial estimated preferences

$\hat{x}_1^o$ and $\hat{x}_2^o$, and we also know the correct overall score $t(o)$. Thus we can estimate weight $\hat{w}_1^o$ estimation from equation

$$t(o) = \hat{w}_1 * \hat{x}_1^o + (1 - \hat{w}_1) * \hat{x}_2^o \tag{17}$$

Because resulting weights can be different for different items, we take the final estimation of the weight $\hat{w}_1$ as the average of computed values over all ranked items. Induced user model $< \hat{i}_1, \hat{i}_2, \hat{w}_1 >$ then defines computed contour polygons $\hat{P}^u_{M_2,h}$ for given $h$.

### 4.4. Metric

Metrics are implemented in spatial database. For given algorithm $\alpha$ (here algorithms $M_1$ and $M_2$) and for given level of preference $h$, we can compute corresponding polygon $P_h^u = [K^1, K^2, L^1, L^2, M^1, M^2, N^1, N^2]$ in the data cube, that represents the contour line $[A, B]$ at the level $h$ in the preference cube (see Figure 5). First we compute the contour line itself. According to the chosen level of preference, some vertexes can merge together and the octagon can become the hexagon or even a tetragon.

$$A = [A_1, A_2] = \begin{cases} [1; h - w_1/(1 - w_1) * (1 - h)], & \text{if } A_2 \geq 0 \\ [h + (1 - w_1)/w_1 * h; 0] & \text{otherwise} \end{cases} \tag{18}$$

$$B = [B_1, B_2] = \begin{cases} [h - (1 - w_1)/w_1 * (1 - h); 1] & \text{if } B_1 \geq 0 \\ [0; h + w_1/(1 - w_1) * h] & \text{otherwise} \end{cases} \tag{19}$$

Next we can compute intersections $X^L$, $X^H$ of horizontal line $[A_1; *]$ with (triangular-shaped) partial preference function $x_1$ and intersections $Y^L$, $Y^H$ of vertical line $[*; B_2]$ with (triangular-shaped) partial preference function $x_2$.

$$X^L = [X_1^L, X_2^L] = [A_1; A_1 * i_1] \tag{20}$$

$$X^H = [X_1^H, X_2^H] = [A_1; i_1 + (MAX_1 - i_1) * (1 - A_1)] \tag{21}$$

$$Y^L = [Y_1^L, Y_2^L] = [B_2 * i_2; B_2] \tag{22}$$

$$Y^H = [Y_1^H, Y_2^H] = [i_2 + (MAX_2 - i_2) * (1 - B_2); B_2] \tag{23}$$

Third we can compute intersections $C, D, E, F$ with (triangular-shaped) preference functions in both dimensions.

$$C = [A_2 * i_2; A_2] \tag{24}$$

$$D = [i_2 + (MAX_2 - i_2) * (1 - A_2); A_2] \tag{25}$$

$$E = [B_1; B_1 * i_1] \tag{26}$$

$$F = [B_1; i_1 + (MAX_1 - i_1) * (1 - B_1); \tag{27}$$

Finally, we can compute boundary of the polygon as follows:

$$K^1 = [E_2; Y_1^H]; \tag{28}$$

$$K^2 = [E_2; Y_1^L]; \tag{29}$$

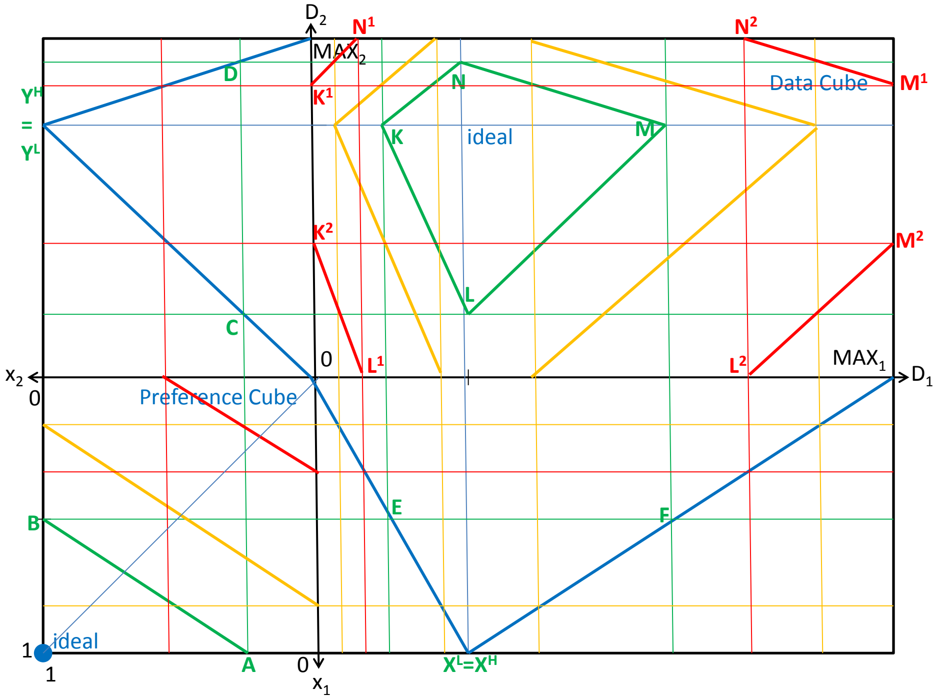


**Fig. 5.** Illustration of polygon computation for given levels of contour lines (green most preferred, yellow medium and red less preferred)

$$L^1 = [X_2^L; C_1] \tag{30}$$

$$L^2 = [X_2^H; C_1] \tag{31}$$

$$M^1 = [F_2; Y_1^L] \tag{32}$$

$$M^2 = [F_2; Y_1^H] \tag{33}$$

$$N^1 = [X_2^H; D_1] \tag{34}$$

$$N^2 = [X_2^L; D_1] \tag{35}$$

Correct ideal points $[i_1; i_2]$ and polygons $P_h^u = [K^1, K^2, L^1, L^2, M^1, M^2, N^1, N^2]$ for $h \in \{0.7, 0.8, 0.9\}$ were pre-computed and stored in *Oracle database* using *Oracle Spatial extension* as MDSYS.SDO_GEOMETRY points and polygon rings. Together with them all computed estimations provided by methods $M_1$ and $M_2$ were stored as well.

## 4.5. Experiments

We have all spatial data stored in the database and indexed by spatial index. This allowed us to effectively compute (not only) areas of both user's correct and computed polygons $P_h^u$, $\hat{P}_{M_1,h}^u$ and $\hat{P}_{M_2,h}^u$ and their intersections.

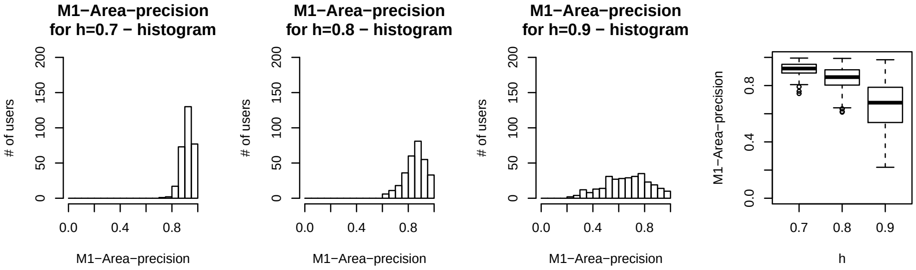


**Fig. 6.** From left to right - histograms of area based precision of algorithm $M_1$ for levels 0.7 , 0.8, and 0.9 and corresponding box-plots.

### Experiments based on the area sizes of polygons and their intersections

Having given level of preference $h$ we can compute both area based precision $AP_h$ and recall $AR_h$ for each user estimation. Formulae, presented above can be computed using SQL expressions

```
$AP_h^u$ =
SDO_GEOM.SDO_AREA(
  SDO_GEOM.SDO_INTERSECTION($P_h^u$,$\hat{P}_{M_k,h}^u$),0.001),
  0.001
  )
/ SDO_GEOM.SDO_AREA($\hat{P}_{M_k,h}^u$),0.001)
```

```
$AR_{M_k,h}^u$=
SDO_GEOM.SDO_AREA(
  SDO_GEOM.SDO_INTERSECTION($P_h^u$,$\hat{P}_{M_k,h}^u$),0.001),
  0.001
  )
/ SDO_GEOM.SDO_AREA($P_h^u$),0.001)
```

Figure 6 represents distribution of area based precision of algorithm $M_1$ for levels 0.7, 0.8 and 0.9 over 300 randomly generated users with different rating frequencies. As the $h$ goes higher, contour polygons become smaller while ideal points (centers) stay on the same places. As a result the area of the polygon intersection decreases faster than the areas of polygons themselves. The more distant are real and estimated ideal points the faster. This results in lower median and higher variance of observed area-precision values.

Figure 7 represents distribution of area based recall of algorithm $M_1$ for levels 0.7, 0.8 and 0.9 over 300 randomly generated users with different rating frequencies. The same reasoning as for area-precision measure leads us to the assumption, that also area-based recall will show lower median and higher variance for higher levels of $h$. This assumption was confirmed by our tests.

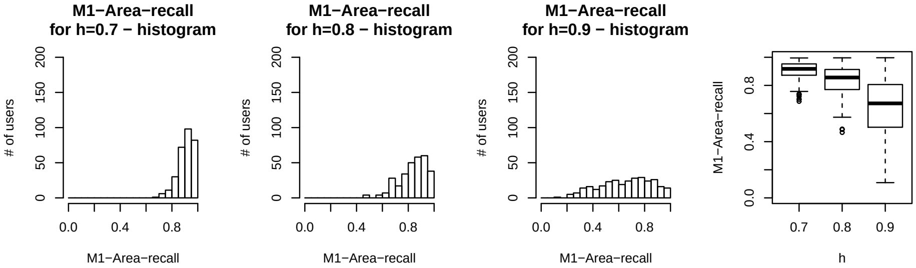


**Fig. 7.** From left to right - histograms of area based recall of algorithm $M_1$ for levels 0.7, 0.8 and 0.9 and corresponding box-plots

Figure 8 on the right shows both area precisions and recalls for individual users for level h=0.7 (red), 0.8 (orange) and 0.9 (green). Every symbol represents one user. We can see, that precision-recall pairs are located near the diagonal. From the computation of both measures we can deduce, that areas of both contour polygons - one belonging to the user and the other to its model are approximately the same. Thus the area of their intersection divided by the area of any of them results in close numbers.

### Experiments based on the number of items in polygons and their intersections

Because area size need not to reflect the successfullness of the method well (some areas of data cube can be empty, while other can contain lot of items, we were interested not only in the area size, but also in number of items in corresponding areas. We can imagine the situation where substantial reduction of intersection area will lead to a small or no change in number of items, located in the intersection because almost all items are located in the remaining area. While the area-recall will be much smaller, the item-recall will remain the same. On the other hand, if almost all existing items would be located out of remaining area, we could notice substantial reduction of item-recall together with marginal reduction of area-recall. Similar considerations apply also to the precision measure. Item based metrics are also computed using *Oracle Spatial extension*, this time as

$IP^u_h$ =

(SELECT COUNT($o$) FROM $Item$ WHERE SDO_INSIDE($o$,

SDO_GEOM.SDO_INTERSECTION($P^u_h$,$\hat{P}^u_{M_k,h}$,0.001))='TRUE')

/

(SELECT COUNT($o$) FROM $Item$ WHERE SDO_INSIDE($o$,

SDO_GEOM.SDO_AREA($\hat{P}^u_{M_k,h}$),0.001)='TRUE')

$IR^u_{M_k,h}$=

(SELECT COUNT($o$) FROM $Item$ WHERE SDO_INSIDE($o$,

SDO_GEOM.SDO_INTERSECTION($P^u_h$,$\hat{P}^u_{M_k,h}$,0.001))='TRUE')

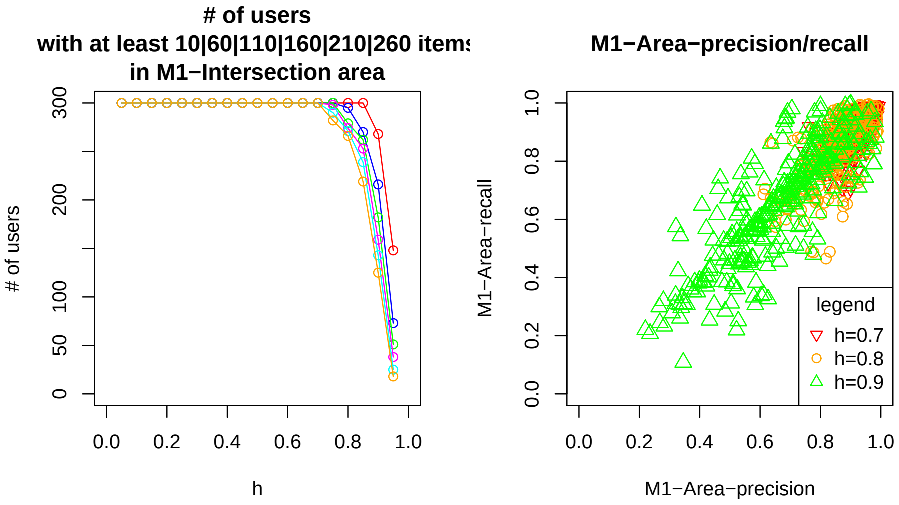


**Fig. 8.** Visualisation of precision and recall values for different users (right). Number of users having at least $n$ items in the intersection of correct and computed contour polygon (left)

```
/
(SELECT COUNT(o) FROM Item WHERE SDO_INSIDE(o,
  SDO_GEOM.SDO_AREA(P_h^u),0.001)='TRUE')
```

Figure 9 represents distribution of item based precision of algorithm $M_1$ for levels 0.7, 0.8 and 0.9 over 300 randomly generated users with different rating frequencies. We can see, that while area-based precision histogram corresponds to Gaussian distribution, where the maximal number of users achieve the average value of the measure, here the distribution is substantially skewed to the right. It means, that while the area of intersection of contour polygons is in average smaller than contour polygons themselves, the intersection contains most of items with almost none located outside the intersection. As the

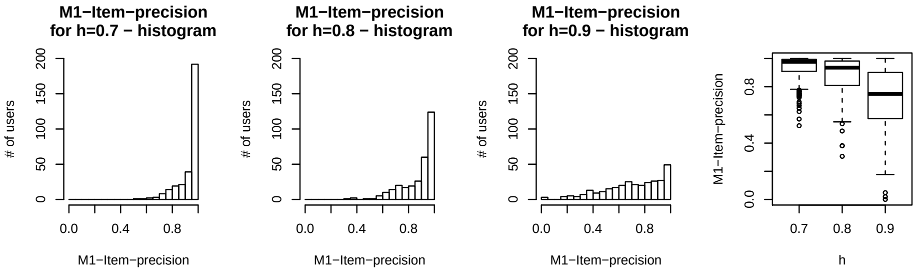


**Fig. 9.** From left to right - histograms of item based precision of algorithm $M_1$ for levels 0.7, 0.8, 0.9 and corresponding box-plots.

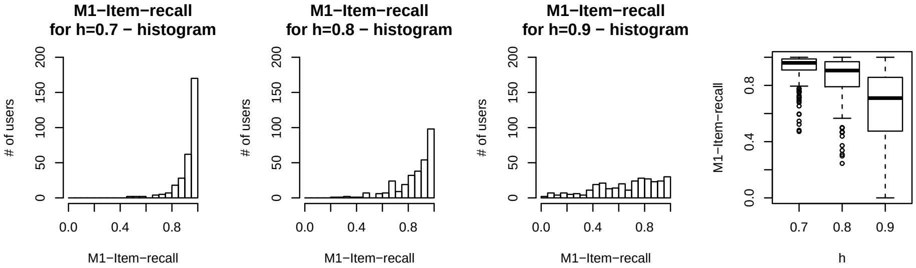


**Fig. 10.** From left to right - histograms of item based recall of algorithm $M_1$ for levels 0.7, 0.8, 0.9 and corresponding box-plots

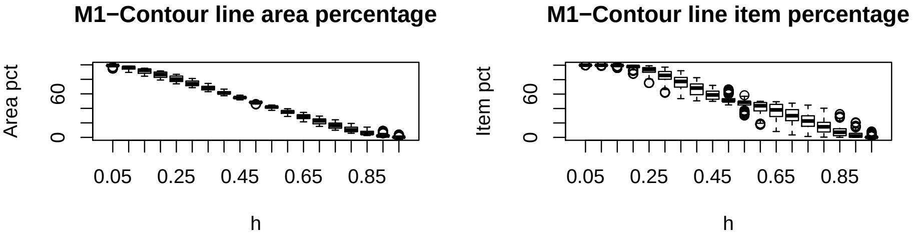


**Fig. 11.** Distribution of percentage of both area size (left) and item number (right) for different levels $h$

$h$ grows the intersection area decreases, and some items become outside the intersection area. It causes gradual decreasing of the item-precision.

Figure 10 represents distribution of item based recall of algorithm $M_1$ for levels 0.7, 0.8 and 0.9 over 300 randomly generated users with different rating frequencies. We can see very similar behaviour as in case of item-based precision. As $h$ increases, some items leave decreasing intersections of contour polygons while remaining in user contour polygon. Again, the item recall for average user will gradually decrease.

We were also interested in how many interesting items with a preference exceeding a certain limit $h$ exist for a given user. Figure 8 on the left shows that only approximately 25 users have less than 10 items in the intersection of correct and computed contour polygon for $h = 0.95$. Approximately one half of users still has 250 and more items in this intersection. The absolute number of items depends on the density of items near the user's ideal point. With less items in the data cube this number will be proportionally smaller.

### Comparison of area- and item- based metrics

Figure 11 shows how the distribution of contour polygon area size and item number within it are affected by increasing required level $h$. It is possible to see, that with higher required level $h$ the average percentage of contour polygon size decreases. With level $h$ close to zero the contour polygon covers almost whole data cube. With level $h$ close to one, the area containing sufficiently preferred items decreases to zero size. For any preference level the box-plot is very small. From this we can see, that contour polygons for all users have almost the same size for given value of preference level $h$. The distribution of object numbers within contour polygons shows higher variances. Depending on whether the user has the ideal point in an area with more or fewer objects the number of items within polygons decreases at some levels faster or slower, than the area size itself. Average area size as well as average item size decrease almost linearly.

### Other algorithms and metrics

So far we have discussed results based on the $M_1$ algorithm and metrics, that need correct contour polygon for each user, preset mesh of pivots, or both. Algorithm $M_2$ doesn't rely on the presence of pivots in the database, and induces all user parameters solely from ranked items. In real environment, exact data about users will be completely unavailable, and thus correct contour polygons will be unreachable. To replace unknown correct contour polygon $P_h^u$, we tried to take convex hull $H_h^u$ of all items $o$ in data cube, that have known user preference $h$ or higher. This approach provided four different combinations of examined algorithm and metric evaluation, as it is shown in Table 1.

**Table 1.** Algorithms and metrics combinations, always need users actions on visited items (either from synthetic data or any other user behavior)

| Computed<br>polygon | Correct<br>user's polygon | Notation<br>in figures |
|---|---|---|
| $M_1$ | Contour polygon $P_h^u$ | M*-P-* needs whole synthetic data |
| $M_2$ | Contour polygon $P_h^u$ | M2-*-P-* needs user's synthetic model |
| $M_1$ | Convex hull $H_h^u$ | M1-*-H-* needs pivots |
| $M_2$ | Convex hull $H_h^u$ | M2-*-H-* can be used without synthetic data |

In general, we can expect that metrics computed using user's correct contour polygon will show better results than corresponding results computed using convex hulls. It is because convex hulls are smaller and represent only subsets of items with required level of preference $h$. I.e. $H_h^u \subseteq P_h^u$. While the user's correct contour polygon can contain tens of sufficiently preferred items, it could happen that the user visited (and rated) for example only three of them. In this case the convex hull $H_h^u$ will be a triangle somewhere inside the contour polygon $P_h^u$. Its size depends on diversity between visited objects. If all of them are located near each other, the triangle will be very small. Moreover, if the level $h$ exceeds rating of the worst of these three items, the convex hull will be constructed by two remaining items, and will have zero area (with current algorithm, some geometric extrapolation is possible to avoid this ([22])). Thus we can expect in general worse results of both area and item-based measures in our tests.

The results obtained by closest neighbour pivot algorithm $M_1$ would be typically better than corresponding results for induced algorithm using centre of mass $M_2$. While the

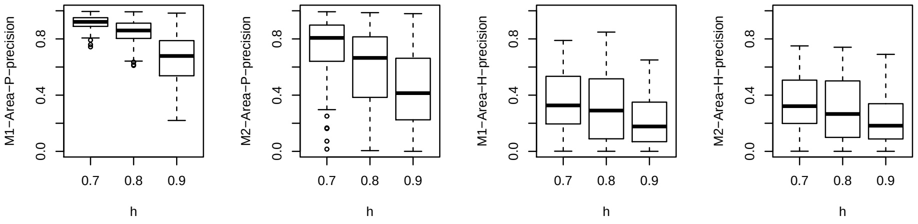


**Fig. 12.** Comparison of Area precisions achieved by different combinations of algorithm and user polygon As we expected, results where the correct contour line were approximated using convex hulls achieved worse values. Compare the third box with the first or the fourth with the second. Also replacing algorithm $M_1$ by $M_2$ show worse results as expected. Compare the second box with the first or the fourth with the third.

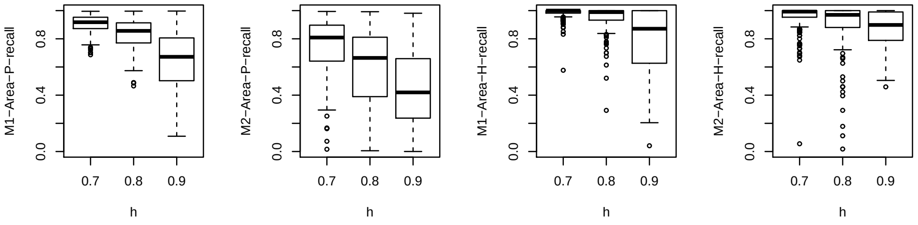


**Fig. 13.** Comparison of Area recalls achieved by different combinations of algorithm and user polygon. Similar comments as in Figure 12 apply.

first algorithm takes the closest available user approximation, the second one can be mistaken, if the user have highly rated only – or almost exclusively – items on one side of his/her ideal point. If user's ideal point $i_k$ is in the half of the dimension, he/she can assign ratings 0.5 to items in the first or third quarter of the dimension. If only items from one group are rated (which is possible because each user visits and rates only small amount of items), their centre of mass and thus the ideal point estimation $\hat{i}_k$ can be significantly shifted to one side. Nevertheless, using $M_2$ is more realistic, in practice it would be difficult to build a set of pivots and compare each user with all of them.

Figure 12 compares precisions achieved by different versions of algorithms and user polygons. Results shown in this figure follows our expectation.

On the other hand similar Figure 13 that compares recalls show much better (higher) values for variants that compute recall using convex hulls $H_h^u$ in comparison with variants that compute the recall using correct contour polygons $P_h^u$. It is caused by the fact, that convex hulls, created with the only knowledge about ranked objects, are much smaller than correct contour polygon. Typically it is fully inside estimated contour line polygon $P^u_{M_k,h}$. Thus the intersection $H_h^u \cap \hat{P}^u_{M_k,h} = H_h^u$ and the recall is equal to 1. If the user rated very small number of items, respectively there are no his/her ratings higher

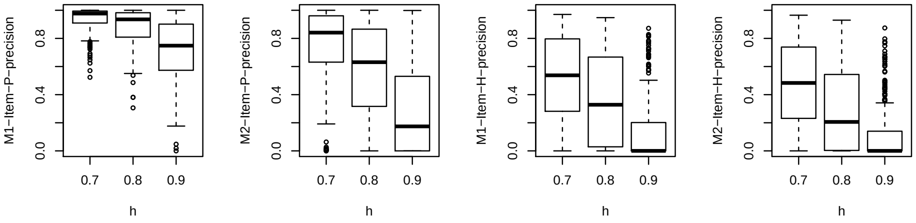


**Fig. 14.** Comparison of Item precisions achieved by different combinations of algorithm and user polygon.

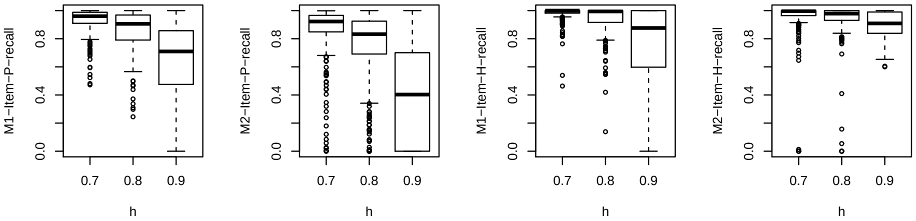


**Fig. 15.** Comparison of Item recalls achieved by different combinations of algorithm and user polygon.

than expected preference level $h$, the convex hull $H_h^u$ cannot be computed at all and both precision and recall for this user cannot be evaluated.

Figures 14 and 15 show similar behaviour. As number of items in given area decreases quickly than the area itself, item-measure results have usually slightly lower median value, greater variance and so among others substantially lower first quartile and minimal values than corresponding area-measures.

Last two figures 16 and 17 compare both achieved Area precision / recall distribution and Item precision / recall distribution for all users. Second plot in Figure 16 shows, that areas of estimated contour polygons $\hat{P}_{M_2,h}^u$ are mostly of the same size as correct contour polygons $P_h^u$, even if they are not perfectly aligned. Thus both $M_2$ Area precision and $M_2$ Area recall are almost the same, and the visualization shows all results on the diagonal. The revision showed, that only 4 percent of users differs in areas of correct and computed contour polygons by more than 5 percent. This shows limitations of our synthetic data and methods. And remains a challenge for future work.

In first two subsections we have described experiments by algorithms. Then we commented results. Much more experiments and comparisons can be made. We hope this has shed light to the nature of proof of concept.

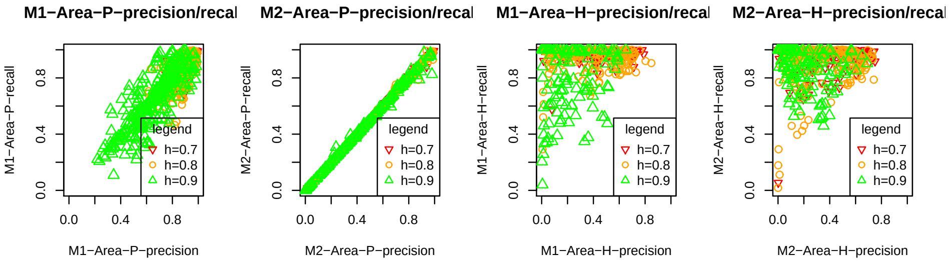


**Fig. 16.** Comparison of Area precisions / Area recalls for individual users achieved by different combinations of algorithm and user polygon. The second figure seems suspicious. It is still correct, as only 4 percent of users differs in areas of correct and computed contour polygons by more than 5 percent. This shows limitations of our synthetic data and methods. And creates a challenge for future work.

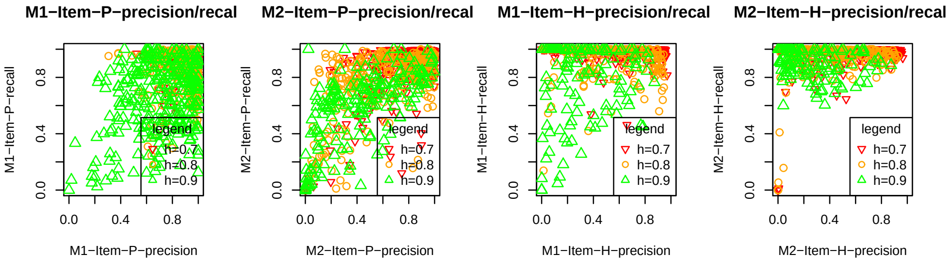


**Fig. 17.** Comparison of Item precisions / Item recalls for individual users achieved by different combinations of algorithm and user polygon.

## 5. Related research

Although our paper presents a formal model and experiments are only on synthetic data, still our main motivation comes from e-commerce.

E-commerce sites try hard to find out what is a user looking for. Various aspects are taken into account - design, user experience, usability etc. To achieve this, various search and filtering techniques are used (see e.g. [10] and many others). Although it is more involved in areas where attributes are not so easy to describe (e.g. clothing), for our motivation value filtering in ordered (numeric) domains is sufficient. Best practices consider filtering by category, by theme, multiple values of the same type, several displays, truncation of more than ten value ... They are advised to review how customers use filters, improve user experience, never return “no results”, care about speed ... Others concentrate on NLP, voice, image, context (attribute), personalized filtering techniques.

In our approach relaxation guarantees, that there are never ”no results”. The linearity of our components increases speed of reply. More values of the same type can be modeled by partly linear preferences consisting of e.g. two triangular or trapezoidal shapes which was not considered here.

Output of e-commerce applications usually use list or grid view. We offer an additional spatial view. Using human ability to grasp overall information is usually used in information visualization of large data according to a fixed metrics. Our visualization is personalized as it uses the closeness notion derived from overall user's preference and corresponding contour line.

Intuitiveness as one of attributes of user experience was also original idea of QBE - querying by example. Namely, an untrained user should be able to specify query without any knowledge of programming. Formal representation of QBE are tableaux queries, see [1]. Our model induces also a form of tableaux model with inequalities (above the contour line). These were introduced in [12] and further studied mainly from complexity theoretic point of view [16]. Once it influenced UX design ([19]), nevertheless today this connection to QBE is no more visible. Maybe our approach can revive this point of view.

One possibility of representing relaxation are fuzzy sets. The "fuzzy world" is not black-and-white, it recognizes degrees of shade. Hence it admits also relaxation of preference, if preference is interpreted as fuzzy score, see [17]. Fuzzy systems were initiated by paper [3]. Fuzzy systems as a tool for combining information (needed for multicriterial querying) were used in [7]. Original motivation was multimedia search in IBM Almaden project Garlic [9], where we can see a graphical querying interface without spatial nor personalized visualization. R. Fagin, A. Lotem and M. Naor in [8] introduced a formal model (FLN) in a more general use case, namely for querying web accessible databases. FLN considers only deductive problem (top-k) and does not touch induction for multiuser personalisation. For us it is most important that [8] introduces preferences for each attribute separately and hence implicitly defines a FLN-class of functions. We simplify these only to linear ones and hopefully win the speed. We extend this FLN approach by considering learning of both attribute preferences and combination function from linear FLN-class of functions.

We used fuzzy sets to model preferences in [17]. We considered learning user preferences for recommender systems in [18].

Another model of relaxing selects appears as acceptable violation of ideal values in AHP - analytic hierarchy process method of T. L. Saaty [20]. Visualization to support AHP process is widely studied, see e.g. [2]. It uses various models like treemaps, parallel coordinates, scatterplot matrices and the tabular visualization ([5]). With a certain degree of simplification we can say that AHP is used for decision support, usually assisted by a human, on low number of objects with very complex (hierarchical) description. In our case we need fully automated fast recommendation on large number of objects with relatively simple (shallow) description.

Challenge response framework was originally motivated in set theoretic study in real analysis (calculus) in [21]. We coined it Galois-Tukey connection. A. Blass ([4]) showed that this idea has appeared in different settings. Namely, it can be used also in computer science reduction of combinatorial problems. He later coined it "challenge-response". We have already mentioned that ChRF facilitates a much more general principle which appear in various types of human endeavor. Notice, that ChRF also appeared as a (http) authentication scheme in [6]. We will call these authentication procedures ChRF in narrow sense. Our approach will be called ChRF in broad sense.

Stochastic data creation with various types of distribution is described in [15]. Extensive experiments with different versions of top-k algorithms (also those from [8]) are provided. We created synthetic (stochastically generated) data set with additional user model with overall preference known and use it for inductive task.

We are aware of a gap between academia and industry, as described e.g. in [11]. Nevertheless we try to avoid proprietary e-commerce solutions. Our formal model based on Fagin-Lotem-Naor [8] class of functions, Challenge-Response-Framework (in broader sense) and visual (spatial) presentation is up to our knowledge unique. Although it is based, so far, only on proof-of-concept experiments on synthetic data, it is a promising candidate worth of future research. Especially when we look for generic models based on sound formal models.

## 6. Conclusions, future work

Main theme of this paper are e-commerce systems and we present an alternative proposal of filtering. Our proposal consists of a formal model integrating some aspects of Fagin-Lotem-Naor approach, Challenge Response Framework (in broad sense) and visual (spatial) aspects. Our main result is a candidate for new generic solutions.

Our proposal was tested on one collection of synthetic data. Of course it does not imply anything for practical applications. On the other hand, as observed by [11] online evaluation in real-world scenarios can be risky for e-commerce. There can appear several problems, such as high resource demands, temporal complexity and the lack of repeatability or potential negative impact on the user experience.

Moreover, companies usually do not share their data, even not historic ones and anonymized for off-line testing. Next step in improving business value of our solutions probably could be creating more synthetic data with distribution (at least statistically) similar to real world data. Especially, it would be interesting to provide tests with large, sparse data and large number of users.

There is another aspect of data we are working with. Our data contain information about user linear FLN-model - ideal points and weights. Let us consider situations where we have only user-item matrix with ranking. Instead of pivots we can use other users behavior in some ”spatial” version of collaborative filtering and models estimated from k-NN users. Our acquaintance with using convex hull in place of ”correct” user model shows that this is usually a subset of ”computed” polygon. Here probably AHP can us to help to consider triangle (trapezoidal, generalized partially linear) preference only in acceptable distance from ideal point or interval. Something like deviation tolerance saying how precipitous relaxation should be. Some initial experiments with ceteris paribus, categorical and nominal data were provided - this is a challenge for future work.

We are convinced that users’ preferences usually depend on low number of attributes. Nevertheless, the two are really the minimum. In computer graphics and information visualization there are techniques for visualization of more complex data. Nevertheless we need to have our visualization personalized and this is based mainly on ability to visualize contour lines (areas or items in those areas). For instance it is difficult to visualize contour lines in parallel coordinates in higher dimensions. We have some preliminary work done already on visualization of four (up to eight) dimensions.

What is also unknown is how our visualization can improve user experience and usability. We did not consider here any aspects of design. To provide user studies in this direction is definitely an interesting task for future.

## References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
2. T. Asahi, D. Turo, Ben Shneiderman. Using Treemaps to Visualize the Analytic Hierarchy Process, Information Systems Research 6,4 (1995) 357–375
3. Bellman, R. E., Kalaba, R. E., Zadeh, L. A.: Abstraction and Pattern Classification, RAND memorandum RM-4307-PR, October 1964, https://www.rand.org/pubs/research_memoranda/RM4307.html
4. Blass, A.: Questions and Answers - A Category Arising in Linear Logic, Complexity Theory, and Set Theory. In: J.-Y. Girard et al. (eds.) Advances in Linear Logic, London Math. Soc. Lecture Notes, vol. 222 pp. 61-81 London Math. Soc. (1995)
5. E. Dimara, A. Bezerianos, P. Dragicevic. Conceptual and Methodological Issues in Evaluating Multidimensional Visualizations for Decision Support. IEEE Transactions on Visualization and Computer Graphics, Institute of Electrical and Electronics Engineers, 2018, 24
6. Encyclopedia of Cryptography and Security. Editors: van Tilborg, Henk C.A., Jajodia, Sushil (Eds.) Springer Science and Business Media 2011
7. Fagin, Ronald, Combining Fuzzy Information from Multiple Systems, J. Comput. Syst. Sci. 58,1 (1999) 83–99
8. Fagin, R., Lotem, A., Naor, M.: Optimal aggregation algorithms for middleware. JCSS 66,4 (2003) 614–656
9. L. M. Haas, R. Fagin, M. D. Flickner, P. Schwarz et all. Querying Multimedia Data from Multiple Repositories by Content: the Garlic https://www.researchgate.net/publication/2764550, downloaded pdf accessed May 29, 2020
10. Holst, Ch.: The Current State Of E-Commerce Filtering, https://www.smashingmagazine.com/2015/04/the-current-state-of-e-commerce-filtering/. Last accessed 8 May 2019
11. D. Jannach, M. Jugovac. Measuring the Business Value of Recommender Systems. ACM TMIS 10,4, Article 16 (December 2019), 23 pages
12. A. Klug. On Conjunctive Queries Containing Inequalities. Journal of the Association for Computing Machinery 35,I (1988) 146-160
13. Kopecky M., Vojtas P. (2019) Graphical E-Commerce Values Filtering Model in Spatial Database Framework. In: Welzer T. et al. (eds) New Trends in Databases and Information Systems. ADBIS 2019. Communications in Computer and Information Science, vol 1064. Springer, Cham, pp 210-220
14. Kopecky, M., Vomlelova, M., Vojtas, P.: Basis Functions as Pivots in Space of Users Preferences. In: Ivanovic, M. et al. (eds.) ADBIS 2016, CCIS, vol. 637, pp. 45–53. Springer (2016).
15. A. Marian, N. Bruno, L. Gravano. , Evaluating top-k queries over web-accessible databases, ACM Transactions on Database Systems 29,2 (2004) 319-362
16. Ron van der Meyden. The Complexity of Querying Indefinite Data about Linearly Ordered Domains. Journal of Computer and System Sciences 54, 113-135 (1997)
17. Peska, L., Eckhardt, A., Vojtas, P.: Preferential Interpretation of Fuzzy Sets in Recommendation with Real E-shop Data Experiments. Archives for the Philosophy and History of Soft Computing No 2 (2015) http://aphsc.org/index.php/aphsc/article/view/32/2
18. Peska, L., Vojtas, P.: Using Implicit Preference Relations to Improve Recommender Systems, J. Data Semantics, 6,1 (2017) 15–30
19. Tony Russell-Rose. Designing Search: Entering the Query. UX Magazine Article No :804, March 14, 2012, https://uxmag.com/articles/designing-search-entering-the-query , last accessed May 25th, 2020
20. Saaty, T. L.: Decision making with the analytic hierarchy process. Int. J. Services Sciences 1,1 (2008) 83–98

21. Vojtas, P.: Generalized Galois-Tukey connections between explicit relations on classical objects of real analysis. Israel Math. Conf. Proc. 6 (1993) 619–643
22. P. Vojtáš and A. Eckhardt, ”Considering Data-Mining Techniques in User Preference Learning,” 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Sydney, NSW, 2008, pp. 33-36

*Received: January 15, 2020; Accepted: September 1, 2020.*

CIP – Каталогизација у публикацији
Народна библиотека Србије, Београд

004

COMPUTER Science and Information
Systems : the International journal /
Editor-in-Chief Mirjana Ivanović. – Vol. 17,
No 3 (2020) - . – Novi Sad (Trg D. Obradovića 3):
ComSIS Consortium, 2020 - (Belgrade
: Sigra star). –30 cm

Polugodišnje. – Tekst na engleskom jeziku

ISSN 1820-0214 (Print) 2406-1018 (Online) = Computer
Science and Information Systems
COBISS.SR-ID 112261644

Cover design: V. Štavljanin
Printed by: Sigra star, Belgrade