Com SIS
Computer Science and Information Systems
Volume 16, Number 2, June 2019

Contents

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

Computer Science and Information Systems
Vol 16, No 2, June 2019

Com SIS

Computer Science and Information Systems

Published by ComSIS Consortium

Volume 16, Number 2
June 2019

ComSIS is an international journal published by the ComSIS Consortium

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 16, Number 2, 2019
Novi Sad

Computer Science and Information Systems

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/

ComSIS

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 2018 two-year impact factor 0.620,
- 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

Computer Science and Information Systems

Volume 16, Number 2, June 2019

CONTENTS

EDITORIAL

This second issue in Volume 16 of the Computer Science and Information Systems journal consists of nine regular articles and extended versions of five papers selected from the 8th International Conference on Web Intelligence, Mining and Semantics (WIMS) which was held in Novi Sad, Serbia, on June 25–27, 2018. As is customary, we gratefully acknowledge all the hard work and enthusiasm of our authors and reviewers, without whom the current issue would not have been possible.

Before turning to the contents of the issue, we have the pleasure of announcing the new impact factors of ComSIS. The new two-year impact factor for 2018 is 0.620, while the five-year impact factor is 0.742.

The regular article section begins with "Majority Vote Feature Selection Algorithm in Software Fault Prediction" where Emin Borandag et al. tackle the problem of identification and location of defects in software projects by isolating the most influential software metrics using various feature rankers. It experimentally is shown that employing most significant metrics as features enhances defect prediction, i.e. classification performance of multiple machine-learning algorithms.

Jiawei Li et al., in "Goal-oriented Dependency Analysis for Service Identification," explore the important aspect of service-oriented architecture systems – service identification. The article considers dependency analysis in the business process management domain, applying a dependency tree featuring the relationships among requirements. The dependency relations are analyzed to create business processes via scenarios comprising requirements and process fragments.

The article "Intelligent Query Processing in P2P Networks: Semantic Issues and Routing Algorithms," by AL Nicolini et al. surveys and discusses the major algorithms for query routing in unstructured P2P networks in which semantic aspects (e.g. provenance, nodes' history, topic similarity, etc.) play a major role. A general comparative analysis is included, associated with a taxonomy of P2P networks based on their degree of decentralization and the different approaches adopted to exploit the available semantic aspects.

"Dimension Reduction and Classification of Hyperspectral Images based on Neural Network Sensitivity Analysis and Multi-instance Learning," authored by Hui Liu et al., addresses two issues regarding hyperspectral image classification: high dimensionality and identification of objects as either a "different body with the same spectrum" or "same body with a different spectrum," making it difficult to maintain the correct correspondence between ground objects and samples. In this respect, the proposed method combines neural network sensitivity analysis with a multi-instance learning algorithm based on a support vector machine to achieve dimension reduction and accurate classification for hyperspectral images.

In “Density-Based Clustering with Constraints,” Piotr Lasek and Jarek Gryz present extensions of classical density-based clustering algorithms, NBC and DBSCAN, allowing specification of instance constraints. Knowledge about anticipated groups can be applied by specifying the so-called must-link and cannot-link relationships between objects or points. Experiments show that instance constraints improve clustering quality with negligible computational overhead related to constraint processing.

Mathias Longo et al., in their article “Reducing energy usage in resource-intensive Java-based scientific applications via micro-benchmark based code refactorings,” examine energy efficiency in Java-based high-performance computing for scientific applications. They revisit a catalog of Java primitives commonly used in scientific programming, or micro-benchmarks, to identify energy-friendly versions of the same primitive. The micro-benchmarks are then applied to classical scientific application kernels and machine learning algorithms. Evaluation shows significant reductions of energy usage at both the micro-benchmark and application levels.

“Product Reputation Mining: Bring Informative Review Summaries to Producers and Consumers,” authored by Zhehua Piao et al. proposes a novel product reputation mining approach based on three points of view: word, sentence, and aspect levels. Aggregating the three scores, the reputation tendency and preferred intensity are measures, and top-k informative review documents about the product are selected. Their experiments show that the method produces more helpful results than the existing lexicon-based approach.

In their article entitled “A Tripartite-Graph Based Recommendation Framework for Price-Comparison Services,” Sang-Chul Lee et al. present a novel application of recommending items to users in price-comparison services. First, it is examined why existing recommendation methods cannot be directly applied to price-comparison services, and then three recommendation strategies are proposed, tailored to price-comparison services: (1) using click-log data to identify users’ preferences, (2) grouping similar items together as a user’s area of interest, and (3) exploiting the category hierarchy and keyword information of items.

To finalize the regular article section, “Logical Filter Approach for Early Stage Cyber-Attack Detection,” by Vacius Jusas et al. considers the problem of early detection of long-lasting cyber attacks, where detailed monitoring of network and system parameters is required to be able to accurately identify the early stages of the attack. The article proposes to consider an attack chain consisting of nine stages, proposing a method to detect early-stage cyber attacks based on attack-chain analysis using hardware implementation of logical filters. Experimental evidence supports the possibility to detect attacks in the early stages.

GUEST EDITORIAL

Special Section on Web Intelligence, Mining and Semantics

We have the pleasure to introduce the special section of the Computer Science and Information Systems journal focused on Web Intelligence, Mining and Semantics.

This special section brings to the reader new research results in the areas of Web Intelligence, Mining and Semantics, with special focus on the synergies between intelligent methods on one side and applications on the other side. We hope that the papers selected for inclusion in this special section will be a valuable resource for researchers and practitioners working in these contemporary research areas.

This special section includes extended versions of selected papers from the 8th International Conference on Web Intelligence, Mining and Semantics (WIMS) held on June 25–27, 2018 in Novi Sad, Serbia. There were 51 submissions to the conference from 30 countries. From the list of accepted papers, 5 papers with high review scores were selected and invited to be extended and submitted to this special section. Finally, after two peer-review rounds, all of them were carefully revised, extended, and improved, and judged acceptable for publication in this special section.

Starting this section is the article "On Approximate k-Nearest Neighbor Searches Based on the Earth Mover's Distance for Efficient Content-Based Multimedia Information Retrieval" by Min-Hee Jang et al., which tackles the problem of too high computational complexity of Earth Mover's Distance (EMD) for multimedia applications by proposing an approximate k-nearest neighbor (k-NN) search method based on EMD. The method relies on the M-tree index structure and post-processing, achieving significant improvement in computational performance with small errors.

The question "How Much Topological Structure Is Preserved by Graph Embeddings?" posed in their article by

Xin Liu et al. is investigated from four aspects: (1) How well the graph can be reconstructed based on the embeddings, (2) The divergence of the original link distribution and the embedding-derived distribution, (3) The consistency of communities discovered from the graph and embeddings, and (4) To what extent can embeddings be employed to facilitate link prediction. It is shown that it is insufficient to rely on the embeddings to reconstruct the original graph, to discover communities, and to predict links at a high precision, meaning that embeddings created by the state-of-the-art approaches can only preserve part of the topological structure.

"Outlier Detection in Graphs: A Study on the Impact of Multiple Graph Models" by Guilherme Oliveira Campos et al. studies the impact of the graph model on outlier detection performance and the gains that may be achieved by using multiple graph models and combining the results. It is shown that assessing the similarity between graphs may be a guidance to determine effective combinations, as less similar graphs are complementary with respect to outlier information they provide and lead to better outlier detection.

Drazen Brdjanin et al., in “Automated Two-phase Business Model-driven Synthesis of Conceptual Database Models” present an approach to automated two-phase business process model-driven synthesis of conceptual database models, based on the introduction of a domain specific language (DSL) as an intermediate layer between different source notations and the target notation, which splits the synthesis into two phases: (1) automatic extraction of specific concepts from the source model and their DSL-based representation, and (2) automated generation of the target model based on the DSL-based representation of the extracted concepts.

Finally, in “Lexicon Based Chinese Language Sentiment Analysis Method,” Jinyan Chen et al. propose a method to identify sentiment in Chinese social media posts, tested on posts sent by visitors of the Great Barrier Reef on the most popular Chinese social media platform Sina Weibo. The article elaborates on the process of capturing weibo posts, describes lexicon construction, and develops and explains the algorithm for sentiment calculation.

We gratefully acknowledge all the hard work and enthusiasm of authors and reviewers, without whom the special section would not have been possible.

Guest editors

Yannis Manolopoulos,
Open University of Cyprus, Cyprus

Mirjana Ivanović,
University of Novi Sad, Serbia

Rajendra Akerkar,
Western Norway Research Institute, Norway

Computer Science and Information Systems 16(2):333–357 https://doi.org/10.2298/CSIS181012005L

A Tripartite-Graph Based Recommendation Framework for Price-Comparison Services*

Sang-Chul Lee[1], Sang-Wook Kim[1], Sunju Park[2], and Dong-Kyu Chae[1]

[1] Department of Computer and Software
Hanyang University, Republic of Korea
{korly, wook, kyu899}@hanyang.ac.kr
[2] School of Business
Yonsei University, Republic of Korea
boxenju@yonsei.ac.kr

Abstract. The recommender systems help users who are going through numerous items (e.g., movies or music) presented in online shops by capturing each user's preferences on items and suggesting a set of personalized items that s/he is likely to prefer [8]. They have been extensively studied in the academic society and widely utilized in many online shops [33]. However, to the best of our knowledge, recommending items to users in *price-comparison services* has not been studied extensively yet, which could attract a great deal of attention from shoppers these days due to its capability to save users' time who want to purchase items with the lowest price [31]. In this paper, we examine why existing recommendation methods cannot be directly applied to price-comparison services, and propose three recommendation strategies that are tailored to price-comparison services: (1) using click-log data to identify users' preferences, (2) grouping similar items together as a user's area of interest, and (3) exploiting the category hierarchy and keyword information of items. We implement these strategies into a unified recommendation framework based on a tripartite graph. Through our extensive experiments using real-world data obtained from Naver shopping, one of the largest price-comparison services in Korea, the proposed framework improved recommendation accuracy up to 87% in terms of precision and 129% in terms of recall, compared to the most competitive baseline.

Keywords: recommendation systems, price-comparison services, random walk with restart.

1. Introduction

Most online shoppers are price sensitive. Since the price of an item may differ from one site to another, the shopper who is looking for a bargain has to visit many shopping sites to compare prices. To save the users' efforts, major portals, such as Google[3], Yahoo![4], Bing[5], and Naver[6], provide a *price-comparison service*.The price-comparison service is useful to the users who know exactly what they are looking for. The user who only has a vague

* Corresponding author: Sang-Wook Kim (wook@hanyang.ac.kr)

[3] Google shopping, http://shopping.google.com
[4] Yahoo! shopping, http://shopping.yahoo.com
[5] bing shopping, http://shopping.bing.com
[6] Naver shopping, http://shopping.naver.com

idea about the item of his interest, on the other hand, still needs to go through numerous items presented by the price-comparison service to narrow their search. A *recommendation system* [1, 21, 34, 38, 46], if provided in conjunction with the price-comparison service, can aid the user in the process of finding preferable item(s).

When a price-comparison site adopts the traditional recommendation systems, it may face several difficulties. If the price-comparison site does not keep record of users' *explicit feedbacks* [28, 30] such as item ratings and purchasing history, it cannot directly utilize the recommendation systems with them. The price-comparison site often suffers from gathering users' explicit feedbacks because it does not sell items but provide link to each shopping mall selling the items. Therefore, we can consider using implicit feedbacks such as click log and search log [19].

The recommendation systems based on implicit feedback, however, cannot produce high-quality recommendation because of the following two distinct characteristics of the price-comparison service. First, the same item may be regarded as different in online shopping, since online shopping sites often use different titles for the same item. This makes it difficult to differentiate whether two users have clicked or searched the same item or different items. Thus, the recommendation system with implicit feedbacks may not be able to correctly compute the similarity [17, 24, 56] between users' preferences. Second, most users utilize the price-comparison service without log-in, and thus the price-comparison service provider cannot collect enough data about the user's history on clicked or searched items. In this situation, the existing recommendation systems suffer from the *cold-start problem* [1, 26, 43, 46], one of the well-known problems of recommendation systems. Because two cold-start users have few items in common, their preferences cannot be compared. If most users are regarded as cold-start users, as in price-comparison services, the recommendation system with implicit feedbacks would produce low-quality recommendation.

In this paper, we propose the strategies for improving the quality of recommendation at price-comparison service sites. First, we use log data for recommendation. Click log is used to identify user's preference and search log is used to filter out previously searched items from recommendation. Second, we group similar items together, and the grouped items are used as a unit of user's preference. Using this strategy, we not only avoid the problem of the same item being regarded as different, but also effectively alleviate the data-sparsity problem. Third, we use similarities between groups to reinforce a user' preference represented by the groups. Since the user tends to prefer the items in the groups similar to the groups that have the items preferred in the past, this strategy can mitigate the cold-start problem.

To adopt the proposed strategies, we need to capture the relationships (1) between users and similar-item groups and (2) between groups. In this paper, we propose a recommendation framework based on a tripartite graph. The proposed framework constructs a graph with nodes corresponding to users, similar-item groups, and groups' features, and links corresponding to the relationships between users and similar-item groups and those between groups and groups' features. Random walk with restart (RWR) [41] on the tripartite graph finds the groups a user is likely to prefer. Then, our framework recommends a set of items to the user from the selected groups.

Through extensive experiments with real-world data, we have verified the superiority of the performance of the proposed framework by comparing it to existing recommendation

methods. We note the following. First, the quality of recommendation of the proposed framework is improved when each additional recommendation strategy is adopted. Second, the recall and the precision of the proposed framework are superior to the existing methods for randomly selected users. Third, the recall and the precision of the proposed framework are also superior to the existing methods for cold-start users. Finally, the user study validates that the proposed framework provides more meaningful recommendations than the existing methods.

The rest of this the paper is organized as follows. Section 2 reviews existing recommendation methods. Section 3 discusses the motivation for a new recommendation system for price-comparison services and proposes three recommendation strategies and the recommendation framework. Section 4 examines the performance of the proposed framework through extensive experiments. Section 5 summarizes and concludes the paper.

2. Related Work

This section briefly reviews several categories of existing recommender systems, including (1) traditional collaborative filtering, (2) group recommendation, (3) implicit feedback based recommendation, and (4) graph-based recommendation using Random Walk with Restart.

The collaborative-filtering approach recommends the items that the users with the tastes and interests similar to the target user liked in the past. The collaborative-filtering approach can be further classified into user-based [6, 9, 23, 27, 29, 44, 47], item-based [22, 37, 45], and graph-based [11, 13, 35, 40, 52, 55] methods. The collaborative-filtering method suffers the cold-start problem, because the taste and interest of a new user can rarely be identified. It also cannot recommend new items which have not yet accrued a sufficient number of ratings.

Recently, several group-recommendation approaches have been proposed [2, 3, 5]. The main goal of these approaches is to maximize the total satisfaction of a target group rather than a single user. Since most of them are based on collaborative filtering while employing some aggregation methods, they also suffer from the inherent shortcomings of collaborative filtering, such as data sparsity and cold-start problems.

Since the above approaches rely on explicit user feedbacks, recommendation is not possible when user-ratings are unavailable. In comparison, some recommendation methods, in particular for web personalization, use 'implicit' click log data or search history data instead of 'explicit' user-ratings [4, 7, 12, 14, 19, 32, 36, 42, 49, 50, 54, 57]. These methods infer the user's preference from the items clicked by the user. They regard the clicks on items as an indirect indication of the user's preference on these items. These approaches could make users free from the burden of providing explicit ratings on items. Also, we do not care whether the ratings are trustable or not [39]. In many cases, however, the quality of recommendation could be unsatisfactory because clicks on items do not always indicate users' preferences [42].

Another research line that are relevant to our work is the *Random Walk with Restart* (RWR). It computes the proximity between a target node and the rest of nodes [41]. The proximity is defined as the probability of staying at each node when random walk through links is performed from a given node with restart. The RWR has been successfully applied to diverse application areas. In the field of recommendation systems, it is known to

provide high accuracy and also known to be useful in the case of analyzing heterogeneous relationships [27, 40]. Like collaborative filtering, we assume that the user would prefer the items clicked by the other users who have clicked on many items clicked by him. Also, the user would prefer the items whose features are similar to those of the items clicked by him. If users, items, item-features, and the relationships among them are modeled as a graph, the RWR can be used to find out the proximity of a user to another users, items, and item-features by reaching each node starting from him. The user would like a user, item or item-feature close to him on the graph even he has never seen before. Thus, we use the proximity to recommend items.

Until now, we have summarized various technologies on recommender systems. However, to the best of our knowledge, there have not been any recommender systems that target the users in price-comparison services. We notice that the following two papers are the most relevant to our research: Lee et al. [31] detected fraudulent users in price-comparison services by analyzing their click logs from the viewpoint of several aspects such as the number of clicks on a single item, a click interval, and a diurnal activity pattern; Gupta et al. [16] performed comprehensive study of analyzing user behaviors and uncovering meaningful characteristics, e.g., "*a user's purchase is highly correlated with the time spent on the site and the search queries s/he wrote before coming on the website*". However, these studies do not provide a working algorithm for recommendation; to the extent of our knowledge, our work is the first one to study a recommender system that works in price-comparison services.

3. Proposed Approach

In this section, we first point out, via preliminary experiments, the problems of using existing implicit-feedback based recommendation methods that infer a user's preference from every single item evaluated or clicked by the user [4, 14, 19]. Then, we present our proposed recommendation framework that successfully remedies the problems and recommends plausible items to users in price-comparison services.

3.1. Motivation

Figure 1 represents the price-comparison process at *Google Product Search*. The user searches items with keyword 'iPhone5,' receives a list of items whose description contains the keyword, and clicks on some of the items on the list to obtain more detailed information or to purchase them. Search and click reveal the user's preference indirectly, and click, in particular, provides a stronger evidence of his preference between the two. Figures 1 (a) and (b) show search log and click log, respectively. From Figure 1 (b), we infer that the user prefers black iPhone 5 to white iPhone5 or iPhone4.

For preliminary experiments, we collected the log data for eight-month periods from Naver shopping, one of the biggest price-comparison sites in Korea. The log contained about 10,000 sampled users and 310,000 items which are clicked at least once by the sampled users. We selected 100 users randomly as target users and produced recommendations for them using a user-based collaborative filtering (user-CF) [45], an item-based collaborative filtering (item-CF) [45], and a graph-based recommendation system (rwr-CF) [15],

Fig. 1. An example of the price-comparison process.

Table 1. An example of the recommendations of the items identical to the previous history

	user-CF	item-CF	rwr-CF
1	safari-style jacket	safari-style jacket	safari-style jacket
2	other jacket	other jacket	safari-style jacket
3	safari-style jacket	safari-style jacket	safari-style jacket
4	safari-style jacket	padding jacket	safari-style jacket
5	padding jacket	safari-style jacket	safari-style jacket
Categories clicked by the target user in the past: safari-style jackets, alpaca wool overcoats, shoulder strap bags, kettles, digital cameras			

respectively. Note that we used the number of clicks on each item as a surrogate for an item rating.

We find two problems from recommendations for the hundred users obtained by the three methods, and the Tables 1 and 2 are their examples, respectively. Table 1 shows

Table 2. An example of irrelevant recommendation or no recommendation

	user-CF	item-CF	rwr-CF
1	2GB USB memory stick	-	-
2	off shoulder knit	-	-
3	character printed T-shirt	-	-
4	backpack	-	-
5	backpack	-	-
Categories clicked by the target user in the past: sofas, tables, sheet papers			

that all three methods recommend the items almost identical to the ones the target user has searched or clicked in the past. Further investigation reveals that the target user in Table 1 has a history of clicking items related to fashion and electronics. Table 2 shows that user-based collaborative filtering recommends the items completely irrelevant to the target user's interest, while the other two methods cannot recommend any items. This phenomenon frequently occurs to cold-start users.

The experiments reveal three potential problems in the existing recommendation methods when applied to price-comparison services: (1) almost identical items are recommended, (2) irrelevant items are recommended, or (3) no item is recommended. These problems happen because of two reasons. First, online shopping malls often describe the same items with slightly different titles, which are regarded as different items by the price-comparison service. As a result, instead of recommending diverse items, the price-comparison service would recommend nearly identical items. Second, because the user is not required to login to use the price-comparison service, the user log keeps the record of the activities during his short session time only. If the record of the target user is limited, the existing methods cannot find users similar to the target user and thus recommend irrelevant items or no item. Note that these problems are unavoidable in recommendation for the price-comparison service. In the following, we propose the strategies to alleviate these two problems.

3.2. Strategies for Recommendation

To provide high-quality recommendation to the user, we propose three strategies for the recommendation systems in price-comparison services. The first strategy is to use log data not only for identifying users' preferences but also for filtering out some items from a recommendation. Generally, the log data is only used for identifying users' preferences. Similarly, we identify the user's preference using clicked items and the number of clicks on them. In addition, we use search log for filtering out the items that are already searched when recommending items to the user.

Although the first strategy may improve the quality of recommendation, the recommendation system with this strategy is still plagued with the problem that the same items may be regarded as different items. The second strategy is to group similar items and use the similar-item group as the unit of user's preference. In traditional collaborative filtering, two users are considered similar if at least one item preferred by each user is

the same. If an individual item is used as the unit of user's preference, the same item is frequently regarded as different in online shopping, and as a result, the similarity between users' preferences may be inaccurately computed. For instance, suppose that item i is listed as i_A and i_B in a price-comparison service, and two users have clicked on i_A and i_B, respectively. Although they practically have similar preference, they would be considered to have different preferences. The second strategy groups similar items using clustering methods[7] and uses the similar-item group as the unit of user's preference. We call the group an *interest-field*.

To solve the cold-start problem, we propose the third strategy of securing additional interest-fields by utilizing similarities between interest-fields. As most users utilize the price-comparison service without log-in, their preferences contain only a small number of interest-fields, which would produce low-quality recommendation, as shown in Table 2. We collect additional interest-fields by including interest-fields similar to those preferred by the user in the past. The similarity of two interest-fields can be measured by the similarity of the descriptions of items in them and/or the closeness of the items in the category hierarchy. Here, the descriptions of items and the categories to which items belong are called *the features of interest-fields*. If we utilize the similarities between interest-fields to secure more interest-fields, the recommendation system would produce higher-quality recommendation even to the cold-start users. We use the three strategies to find the interest-fields that the target user may prefer. After finding the interest-fields, we should recommend individual items from the interest-fields. In the next section, we explain how to practically adopt the three strategies and how to select the individual items in detail.

3.3. The Recommendation Framework

Our recommendation framework consists of four steps, as shown in Figure 2. The first three steps are performed offline, and the last is done online at the time of item search. At Step 1, the framework groups a set of similar items together. At Step 2, it constructs a tripartite graph using the relationships among users, items, and item-features. At Step 3, it identifies the user's preference on items by performing the RWR on the tripartite graph. Note that these three steps can be performed beforehand. At Step 4, the framework recommends a set of items based on the preferences identified in Step 3 in response to the user search.

For Step 1, we use the lowest level in the category hierarchy as the item group, and call it an 'interest-field.' We believe it is more practical and economical to use the category hierarchy which has been already well-organized by portals than employing some clustering methods, such as k-means.

For Step 2, we have developed a tripartite-graph based recommendation framework. The graph consists of three types of nodes: users, interest-fields, and features. The user node is the user who has clicked at least one item. The interest-field node is a group of identical or similar items. The feature node is either the keyword in the item descriptions or the name of a higher-level category.

There exist two types of links. The link between the user node and the interest-field node indicates that the user has clicked some items in the interest-field. The link between

[7] Classification by domain experts can also be used.

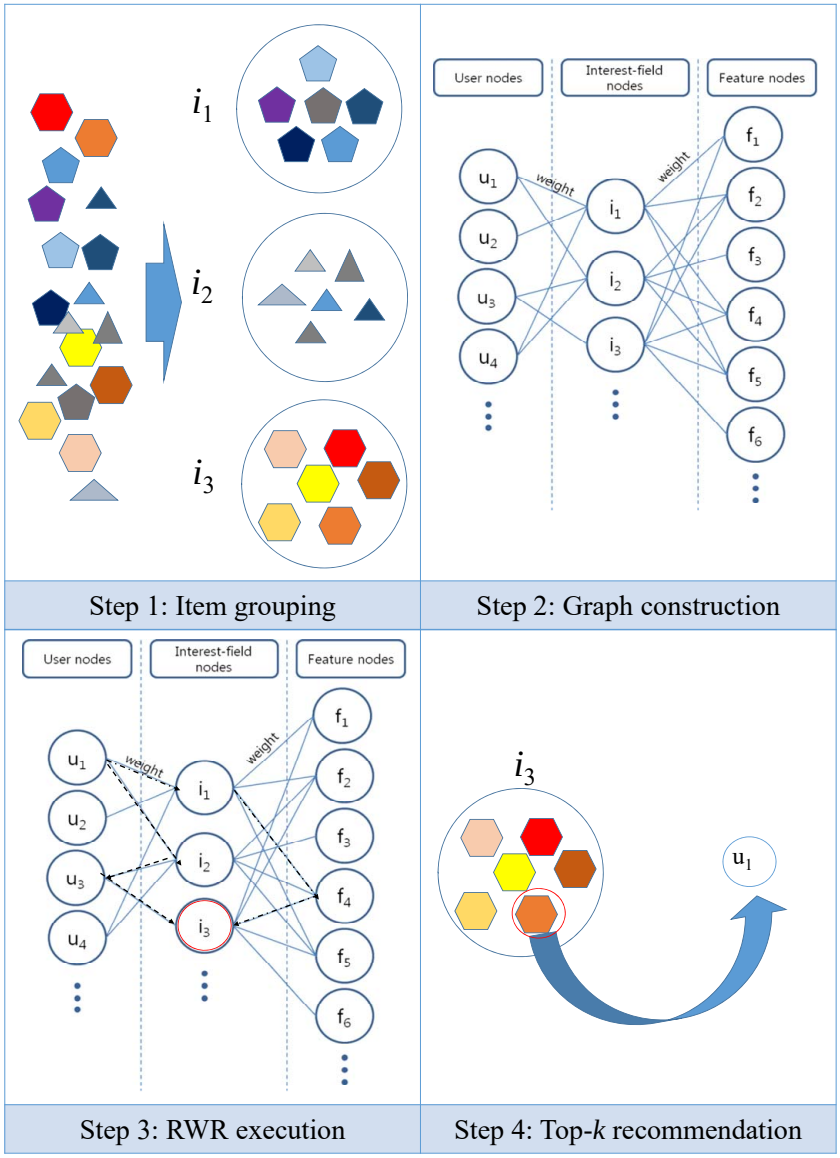


Fig. 2. The detail steps in the proposed framework.

the interest-field node and the feature node captures the fact that the item in the interest-field is described by the keyword represented by the feature node or belongs to the higher category represented by the feature node. Figure 3 represents the tripartite graph of the proposed framework. How to assign the weight of these types of links is explained in detail in the next section.

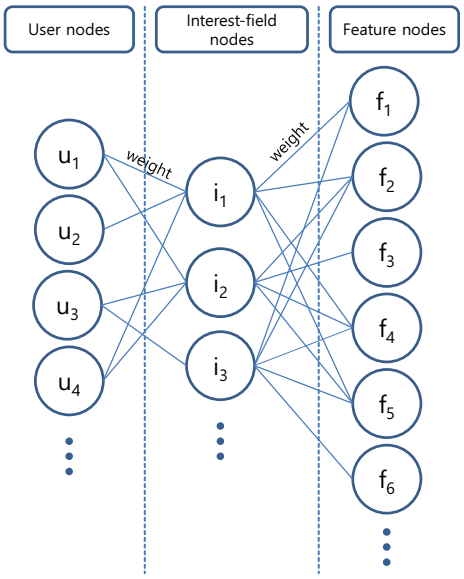


Fig. 3. The tripartite graph used in the proposed framework.

For Step 3, our framework employs Random Walk with Restart (RWR). Equation 1 describes the RWR process. In Equation 1, matrix A is the adjacency matrix representing the tripartite graph. The size of the matrix is the number of all nodes by the number of all nodes, and each cell represents the weight of each link. There exists no link between two nodes of the same type and its value is 0. The vector R_i is the proximity vector of the user where each element is the proximity value of the node on the graph at the i-th step. The initial proximity vector R_0 is set 1 for the node corresponding to the target user, otherwise set 0. The vector E is the restart vector and set as the same as the initial proximity vector R_0. The restart vector is needed to recursively restart from the target user node. The first part of Equation 1 is the random walk to each node in a graph from the target user node, and the second part represents the restart from the target user node. The probability α between two parts of RWR is normally set as 0.85 /citeKon09, Onu09. RWR is recursively computed until the vector R_i converges.

$$R_{i+1} = \alpha A R_i + (1 - \alpha) E \tag{1}$$

Because of the large size of adjacency matrix A, it is infeasible to directly compute RWR at runtime. For instant online recommendation, the framework pre-computes the proximity vectors R represented users' preferences for all users using the fast RWR [48]. Equation 1 represents the 'converged' proximity vector R. In order to compute R, Equation 1 is transformed into Equation 2. In Fast RWR, the proximity vectors of all users are computed without recursion if the inverse matrix of $(I - \alpha A)$ is known in advance. Since the inversion of a large matrix is difficult to compute, we use the technique of partitioning the original graph with Metis and combining the inversions of small matrices [25]

$$R = (1 - \alpha)(I - \alpha A)^{-I} E \tag{2}$$

Using the proximity vector R, the framework selects the interest-fields with high proximity values. Even though the proximity value of an interest-field is high, if the user has already clicked on several items in that interest-field, he would not want to see more items from it. The framework filters out the interest-fields that have received more than a pre-set number of clicks before recommending individual items.

For Step 4, we suggest the following criteria. First, the item that has been clicked or searched is filtered out from recommendation. Exclusion of these items is justified because either the target user has already seen the detailed information of clicked items or he has ignored the items searched but not clicked on purpose. Second, the popularity and/or recentness of items should be considered. The popularity of an item is captured in the total number of clicks it has received, and its recentness is found by the release date. Different price-comparison service providers may use different policies in using the popularity and recentness criteria. For instance, if the provider considers popularity more important, it may recommend an item with a higher number of clicks. On the other hand, if the provider considers recentness more important, it may recommend an item with more recent release date, which can alleviate the latency problem.

Next, we examine the time complexity of each step in order to understand our framework in the performance perspective. In step (1), for having a set of similar items grouped together, we just employ the category hierarchy, which has already been organized by domain experts. So, the time complexity of this step is O(1). In step (2), the time complexity of constructing a tripartite graph is O(e), where e indicates the number of edges in a graph. In step (3), the time complexity of performing RWR on a tripartite graph is again O(e). In step (4), for top-k recommendation, we need to sort all the interest fields according to their RWR scores. Thus, its time complexity is O(I log I), where I indicates the number of interest fields.

3.4. Link Weights

The weight of a link captures the degree of either how much a user prefers an interest-field or how closely related an interest-field and a feature are. Figure 4 shows an example of two types of links connected to a single interest-field node. Here, the feature node is classified into two types: category feature node and keyword feature node.

The number of clicks from user a to interest-field x is used as the weight of link $L(u_a, i_x)$. The link is weighted according to the number of clicks. Unlike the relationship between a user and an interest node, there is little evidence how important a link between an interest-field and a feature node is. In the case of the *category* feature node, the framework assigns different weights for different levels in category hierarchy. For example, Naver Shopping categorizes an item into four levels. Each interest field (i.e., the lowest-level category) is connected to three upper-level category feature nodes, and the weights of those links are set by parameters s, m, and l, where s, m, and l are small, medium, and large levels of product category hierarchy, respectively. On the other hand, since it is difficult to determine which keywords are more important to an interest-field, the same weights are assigned for the links between an interest-field and the *keyword* feature nodes.

Even with the weights assigned, we still face the problem of determining which type of link is more important. Suppose that user u_1 clicks 10 times on interest-fields i_1 and i_1 has 3 keywords. That is, $w(L(u_1, i_1))$ is 10, and $w(L(i_1, f_4))$is $1/3$. We still need to determine the relative ratio of these two links.

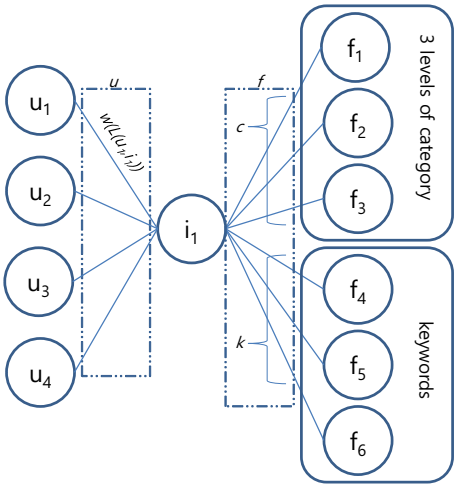


Fig. 4. An example of the weight of links centering an interest-field.

The framework is designed to control the relative importance of two types of links, $L(u,i)$ and $L(i,f)$, using parameters u and f. Equation 5 represents the ratio between two links. In Equation 5, U_x is the set of users who have clicked items in interest-field x, and F_x is the set of the features of interest-field x. When $u:f$ equals to 1:0, the framework is the same as the graph-based collaborative filtering method. When u:f is close to 0:1, the framework is similar to the content-based recommendation method. In this regard, the framework is one of the hybrid methods.

$$\sum\nolimits_{u_a \in U_x} w(L(u_a, i_x)) : \sum\nolimits_{f_b \in F_x} w(L(i_x, f_b)) = u : f \tag{3}$$

The relative ratio between category feature nodes and keyword feature nodes is determined in a similar way, using parameters c and k. Equation 6 represents the ratio between the link from an interest-field node to a category feature node and the link from an interest-field node to a keyword feature node. In Equation 6, FC_x is a set of the category feature nodes of interest-field node x, and FK_x is a set of the keyword feature nodes of x.

$$\sum\nolimits_{f_a \in FC_x} w(L(i_x, f_a)) : \sum\nolimits_{f_b \in FK_x} w(L(i_x, f_b)) = c : k \tag{4}$$

4. Evaluation

In this section, we demonstrate through various experiments that our proposed framework is superior to the existing methods, such as user-based collaborative filtering (user-CF) [6], item-based collaborative filtering (item-CF) [45], user-based one class collaborative filtering (user-OCCF), item-based one class collaborative filtering (item-OCCF) [42] and graph-based recommendation (RWR) [15]. Actually, we found more algorithms in the context of hybrid recommendation (e.g., *Collaborative Topic Regression* (CTR) [51], *Collaborative Deep Learning* (CDL) [53], and SVDfeature [10]) in the literature. However, we could not include them as our baselines since they cannot be directly applied to our content

data (i.e., the hierarchy of product categories and the keywords in the item descriptions), and their extension to fit our data is non-trivial and extremely difficult; we leave this task for our future work.

4.1. Experimental Setup

For experiments, we used the search and click log data of Naver Shopping. Each data point in the search log is composed of the user, the search keywords, and the list of searched items. Each data point in the click log is composed of the user and the list of clicked items. The search log consists of 10,000 users and 9,099,698 items, and the click log consists of 9,997 users and 310,841 items. Since we do not have any login information about users, we regard each session, created when a user visits the shopping mall site, as an identical user. As shown in Figure 5, Naver shopping classifies items into four-level category hierarchy: large, medium, small, and detail category levels. Each level consists of 19, 231, 1461, and 4282 categories, respectively.

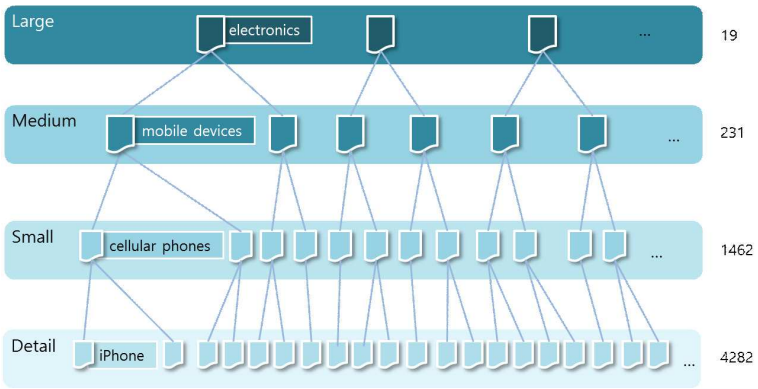


Fig. 5. The Product Category Hierarchy of Naver Shopping.

Before comparing the performance of our framework against other existing methods, we performed two sets of experiments. The first set of experiments was to determine the parameters for the proposed framework (experiment 1). The second set of experiments was self-evaluation, analyzing the performance of the proposed framework by successively adding the second and the third strategies (experiment 2). The performance-comparison experiments with other methods were performed with general users (experiment 3) and cold-start users (experiment 4). We also conducted a user study (experiment 5).

For self-evaluation (experiments 1 and 2) and experiment 3, we randomly selected 100 users among those with the history of clicks on more than 5 detail categories. For each user, a subset of detail categories clicked was randomly selected and used as a test set: 1 category for the users with clicks to 5 to 10 categories, 2 categories for the users with clicks to 10 to 20 categories, and 3 categories for the users with clicks to more than 20 categories. The test set was excluded from the training data set. The evaluation was based on whether a recommendation method was able to recommend the detail categories in the test set. Each recommendation method was evaluated at the small category level as well.

That is, we also evaluated whether the recommendation methods can recommend the detail categories that are the siblings of the ones in the test set.

For experiment 4, we could not evaluate recommendations for real cold-start users because most of them were anonymous users without log-in. Even if some of them were not anonymous users, we still could not find out profiles of them because of privacy issues. So, we generated cold-start users from 706 users who had originally clicked 30 to 80 categories. For each user, five randomly-selected categories were used as a training set and the rest were used as a test set. In addition to the two evaluations mentioned above, we also investigated whether the increase in the number of categories the users had clicked had an impact on the quality of recommendation.

We used three metrics: recall, precision, and coverage, as defined below [18]. The correct-answer items are defined as the items in the test set. Recall is the ratio of recommended correct-answer items to all correct-answer items. Precision is the percentage of recommended correct-answer items to all recommended items. Finally, the coverage is the percentage of users who received recommendation. The coverage metric is used in experiment 4 only, where many cases with no recommendation are reported.

$$recall = \frac{|\{correct{-}answer\ items\} \cap \{recommended\ items\}|}{|\{correct - answer\ items\}|} \tag{5}$$

$$precision = \frac{|\{correct{-}answer\ items\} \cap \{recommended\ items\}|}{|\{recommended\ items\}|} \tag{6}$$

$$coverage = \frac{\#\ of\ target\ users\ received\ the\ recommendation}{the\ number\ of\ target\ users} \tag{7}$$

For experiment 5, we conducted a user study with seven volunteers. Ideally, real users should be judging the quality of the items recommended by each method. It is difficult, however, to identify real users because of the privacy issue for gathering user profiles. Instead, each volunteer manually selected top five users or a single user similar to himself, and evaluated whether the recommendations were useful or not.

4.2. Experimental Results

Experiment 1: Parameter settings for Naver Shopping In this section, we conducted a set of experiments to determine the optimal parameter values for Naver Shopping. The proposed framework should set the weight-ratio parameters: $u : f$ for two types of relationships between users and interest-fields and between interest-fields and features, $c : k$ to handle the importance of links to two types of features, and $s : m : l$ to reflect the levels of the category hierarchy. Table 3 shows the parameter values tried in the experiments. In Table 3, the boldfaced and underlined numbers are the default values for the respective parameters. For example, when we tried out different settings for $u : f$, we fixed the values of $c : k$ and $s : m : l$ to 2:1 and 1:1:1, respectively.

In each experiment, the number of recommended categories was either 10 or 20, and the recommendation was given at the detail level or at the small level. The accuracy of recommendation was measured by recall and precision. Tables 4, 5, and 6 show the accuracy of the recommendation with different parameter settings. The highlighted parameter value shows the highest accuracy. Because an answer set for each user consists of one to three categories, the precision is at most 0.3 where 10 detail-level interest-fields are

Table 3. Parameter settings

Parameters	Values
$u : f$	1:4, 1:3, 1:2, **1:1**, 2:1, 3:1, 4:1
$c : k$	1:0, 3:1, **2:1**, 1:1, 1:2, 1:3, 0:1
$s : m : l$	**1:1:1**, 3:2:1, 5:3:1, 4:2:1, 9:3:1

recommended for the user who has three categories as his answer set. The maximum precision can be lower than 0.3 since most users have clicked less than 20 categories. As shown, the best parameter setting for Naver Shopping is found to be $u : f = 1 : 1$, $c : k = 2 : 1$, and $s : m : l = 1 : 1 : 1$. The only exceptions are the cases where 20 detail-level interest-fields are recommended. In practice, most providers want to recommend far less than 10 items, and thus this exception should not pose any real problem. Also, the best accuracy is obtained when the ratio of c and k is 2:1 in Table 5. The relative importance of the category compared to keywords is because the category is organized well by domain experts, while the keyword is formulated by normal users, not experts. In the following experiments, the best parameter values, $u : f = 1 : 1$, $c : k = 2 : 1$, and $s : m : l = 1 : 1 : 1$, are used.

Table 4. The accuracy of recommendation while changing $u : f$

(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)		Accuracy (small level)	
$u : f$	Precision	Recall	Precision	Recall
1:4	0.018, **0.014**	0.107, **0.172**	0.035, 0.021	0.207, 0.247
1:3	0.014, 0.010	0.088, 0.132	0.025, 0.018	0.165, 0.227
1:2	0.013, 0.011	0.093, 0.158	0.031, 0.020	0.205, 0.253
1:1	**0.022**, 0.012	**0.142**, 0.157	**0.037**, **0.022**	**0.238**, **0.277**
2:1	0.017, 0.011	0.098, 0.127	0.029, 0.019	0.183, 0.230
3:1	0.012, 0.007	0.085, 0.090	0.022, 0.017	0.145, 0.208
4:1	0.010, 0.008	0.072, 0.103	0.023, 0.018	0.160, 0.237

Experiment 2: Self-evaluation In this section, we demonstrate the improvement in accuracy by applying the second and the third recommendation strategies successively to the first strategy of using log data (i.e., implicit-feedback based recommendation)[8].

In the first set of experiments, we compared the recommendation method using user & item bipartite graph, and the one using user & interest-field bipartite graph. The former is the default method, while the latter utilizes the second strategy that adopts the concept of interest-fields to the existing graph-based method. Since the former recommends items

[8] Since Naver shopping does not collect the user's explicit feedback such as the item ratings, the first strategy of using log data is set as the default strategy.

Table 5. The accuracy of recommendation while changing $c : k$

(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)		Accuracy (small level)	
$c : k$	Precision	Recall	Precision	Recall
1:0	0.018, 0.012	0.125, 0.150	0.032, 0.020	0.202, 0.247
3:1	0.015, 0.009	0.093, 0.100	0.031, 0.021	0.183, 0.258
2:1	**0.022**, **0.012**	**0.142**, **0.157**	**0.037**, **0.022**	**0.238**, **0.277**
1:1	0.019, 0.012	0.113, 0.133	0.029, 0.018	0.180, 0.218
1:2	0.015, 0.010	0.103, 0.132	0.023, 0.018	0.147, 0.213
1:3	0.016, 0.010	0.105, 0.127	0.028, 0.020	0.172, 0.228
0:1	0.015, 0.009	0.100, 0.113	0.023, 0.019	0.143, 0.223

Table 6. The accuracy of recommendation while changing $s : m : l$

(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)		Accuracy (small level)	
$s : m : l$	Precision	Recall	Precision	Recall
1:1:1	**0.022**, 0.012	**0.142**, 0.157	**0.037**, **0.022**	**0.238**, **0.277**
3:2:1	0.015, 0.011	0.093, 0.130	0.028, 0.019	0.170, 0.237
5:3:1	0.012, 0.010	0.080, 0.120	0.022, 0.016	0.135, 0.177
4:2:1	0.021, **0.013**	0.135, **0.170**	0.033, 0.022	0.208, 0.257
9:3:1	0.013, 0.009	0.083, 0.112	0.026, 0.017	0.157, 0.157

while the latter recommends interest-fields, we used the lowest-level categories that containing the items recommended by the former for comparison. Table 7 shows the recall and precision with 10 or 20 recommendations by two methods. The results show that the second strategy performed better and confirm our claim that using interest-fields is more advantageous to identify user's preference than using items.

Table 7. The accuracy comparison when applying the second strategy: (a) user & item bipartite and (b) user & interest-field bipartite

(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)		Accuracy (small level)	
Graphs	Precision	Recall	Precision	Recall
(a)	0.008, 0.006	0.057, 0.080	0.017, 0.013	0.090, 0.128
(b)	0.011, 0.007	0.083, 0.103	0.019, 0.014	0.132, 0.182

In the second set of experiments, we compared the graph-based method with user & interest-field bipartite graph and the one that uses the features of interest-fields. The former

is based on the second strategy, while the latter utilizes the third recommendation strategy in addition. Three cases with different features were analyzed: using category-hierarchy only, using keyword only, and using both category hierarchy and keyword. As shown in Table 8, the method with both category hierarchy and keyword information exhibited the highest accuracy. Compared to the methods using a single feature, the method using both category hierarchy and keyword information can secure more interest-fields that are not clicked but are likely to be preferred by a user. In the following experiments, we use the graph-based method with a tripartite graph using both features.

Table 8. The accuracy comparison when applying the third recommendation strategy: (a) User & interest-field bipartite, (b) tripartite (category hierarchy), (c) tripartite (search keyword) and (d) tripartite (both)

(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)		Accuracy (small level)	
Graphs	Precision	Recall	Precision	Recall
(a)	0.011, 0.007	0.083, 0.103	0.019, 0.014	0.132, 0.182
(b)	0.018, 0.012	0.125, 0.150	0.032, 0.020	0.202, 0.247
(c)	0.015, 0.009	0.100, 0.113	0.023, 0.019	0.143, 0.223
(d)	0.022, 0.012	0.142, 0.157	0.037, 0.022	0.238, 0.277

Experiment 3: Cases with general users In this set of experiments, we compared the performance of the proposed framework with three existing methods (rwr-CF, user-CF, item-CF, user-OCCF, item-OCCF) for general warm-start users. Since the existing methods recommend items, for fair comparison, we let the existing methods recommend items until the number of distinct interest-fields reaches 10 or 20.

Table 9 shows the results of the recall and precision of each method with different category-levels for evaluation. It is observed that the proposed framework performed better than the existing methods in terms of accuracy. Since the accuracy of user-OCCF at the detail category level is too low, the results of user-OCCF at the detail level are not reported in Table 9. At the detail category level, the proposed framework improved precision by 77% to 150%, recall by 54% to 119%, and F-measure by 134% to 223% over the existing methods when 10 interest-fields were recommended, and improved precision by 19% to 90%, recall by 5% to 68%, and F-measure by 74% to 89% when 20 interest-fields were recommended. At the small category level, the proposed framework improved 18% to 87% of precision, 48% to 129% of recall, and 69% to 92% of F-measure when 10 interest-fields were recommended, and improved 0% to 48% of precision, 27% to 87% of recall, and 29% to 51% when 20 interest-fields were recommended.

These improvements come from two factors. First, interest-fields have an advantage over individual items when identifying user's preference. While the existing methods

do not classify two users as similar if they have never clicked identical items but have clicked similar items, they are regarded similar when interest-fields are used as the unit of recommendation. Second, the use of the features of interest-fields makes it possible to recommend interest-fields, even if a target user has clicked a very small number of items. While the existing methods provide the user with irrelevant recommendation or no recommendation, using the features of interest-fields, the proposed framework can secure more interest-fields that have not been clicked by the user but have the features similar to the interest-fields that he is already interested in.

Table 9. The accuracy comparisons between our approach and the existing methods

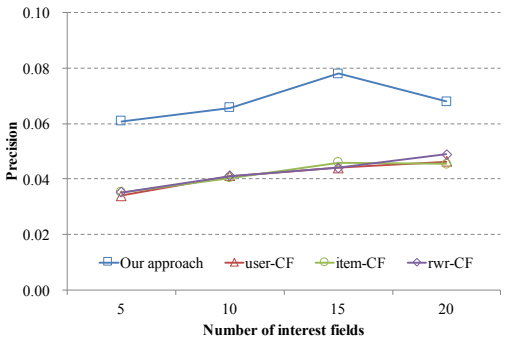
(10 interest-fields, 20 interest-fields)

level	Accuracy (detail level)			Accuracy (small level)		
Methods	Precision	Recall	F-measure	Precision	Recall	F-measure
Our approach	0.022, 0.012	0.142, 0.157	0.038, 0.022	0.037, 0.022	0.238, 0.277	0.064, 0.0340
rwr-CF	0.009, 0.006	0.066, 0.093	0.016, 0.0120	0.020, 0.015	0.104, 0.148	0.033, 0.026
user-CF	0.012, 0.010	0.092, 0.150	0.012, 0.012	0.032, 0.022	0.162, 0.219	0.038, 0.026
item-CF	0.009, 0.007	0.065, 0.108	0.012, 0.013	0.023, 0.017	0.120, 0.191	0.038, 0.031
user-OCCF	- , -	- , -	- , -	0.002, 0.001	0.002, 0.003	0.003, 0.003
item-OCCF	0.000, 0.000	0.004, 0.007	0.001, 0.001	0.002, 0.001	0.017, 0.028	0.003, 0.003

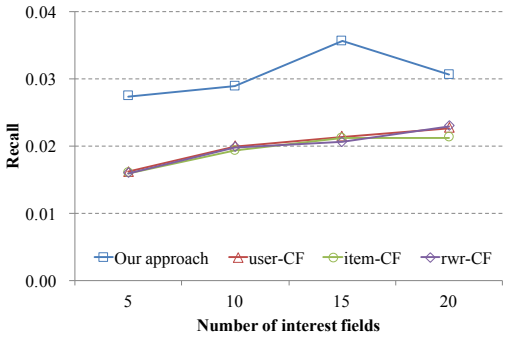
Experiment 4: Cases with cold-start users In this set of experiments, we compared the performance of the proposed framework with rwr-CF, user-CF, item-CF methods for cold-start users. Note that all OCCF based method perform too low accuracy to show in the figures. So, we decide to exclude them in this section. Figure 6 shows the change in accuracy with the increase in the number of interest-fields clicked by each target user. In Figures 6 (a), (b), and (c), the x axis represents the number of the detail categories clicked by each target user, and the y axis represents the precision and recall with 10 recommendations. The proposed framework improved precision by 46% to 79%, recall by 30% to 70%, and F-measure by 35% to 74% over user-CF that showed the best performance among the existing methods. Note that the accuracy of the proposed framework dropped slightly when the number of clicked interest-fields increased from 15 to 20. We conjecture that the cold-start user turns into a general user by that time, so there is little improvement in accuracy even if he clicks more interest-fields.

At the detail category level, the proposed framework improved precision by 77% to 150%, recall by 54% to 119%, and F-measure by 134% to 223% over the existing methods when 10 interest-fields were recommended, and improved precision by 19% to 90%, recall by 5% to 68%, and F-measure by 74% to 89% when 20 interest-fields were recommended. At the small category level, the proposed framework improved 18% to 87% of precision, 48% to 129% of recall, and 69% to 92% of F-measure when 10 interest-fields were recommended, and improved 0% to 48% of precision, 27% to 87% of recall, and 29% to 51% when 20 interest-fields were recommended.

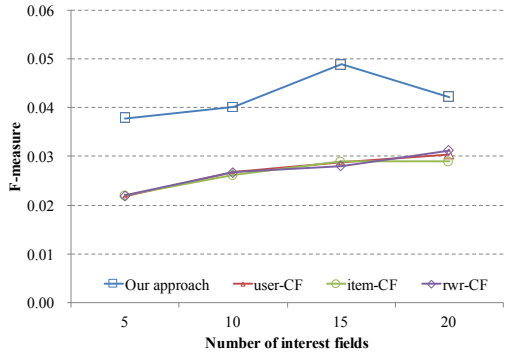
Figure 7 shows the result of the coverage, i.e., how many users received the recommendation by each method. The proposed framework provided all users with recommendation;



(a) Precision



(b) Recall



(c) F-measure

Fig. 6. The accuracy comparison of the recommendations for cold-start users.

while the existing methods were not able to produce recommendation for 25 to 35% of users when they clicked on 5 detail categories. We note that both user-CF and rwr-CF had similar coverage of users. The users who had received recommendation by each method overlapped significantly. All existing methods were getting close to 100% of the coverage with the increase in the number of detail categories. It is important to produce the

recommendation for cold-start users, and thus the proposed method is more suitable for price-comparison services.

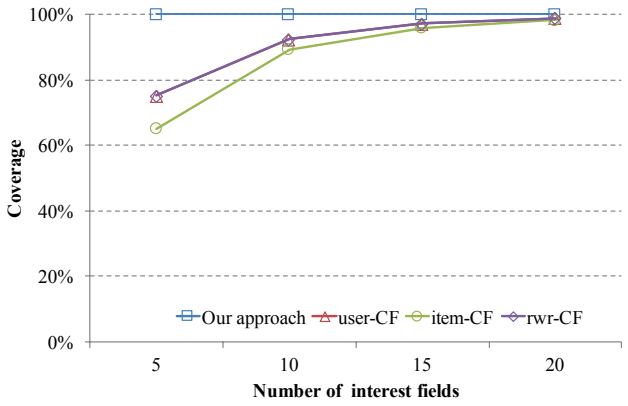


Fig. 7. The coverage comparison of the recommendations for cold-start user.

Experiment 5: The user study In this final set of experiments, we conducted user studies by asking seven volunteers to evaluate the effectiveness of recommendations obtained from six methods (our approach, user-CF, item-CF, rwr-CF, popularity-based recommendation, and random recommendation). Each volunteer went through 100 users who were randomly selected in experiment 3 examined their clicked detail categories, and selected five users who were most similar to himself. Then, he evaluated the recommendations by six different methods for these five users. He gave binary scores to the interest fields recommended by six methods, indicating whether each recommendation was useful for him or not.

Table 10 presents the result of the effectiveness with top five similar users. The boldfaced number in a gray cell represents the method each volunteer selected as the best. Four out of seven volunteers evaluated that the quality of the recommendation by the proposed framework was the best. The average and the trimmed average scores of the proposed framework are higher than the others. Here, the trimmed average means the average while excluding the highest and the lowest scores. Table 11 summarizes the result of the effectiveness of the recommendation methods with the user most similar to each volunteer. The result is similar to that of Table 10.

Note that in real world the price-comparison service provider tends to recommend popular interest-fields which have been clicked many times by many users and these recommendations is evaluated as satisfactory. We expect an improvement in user satisfaction when applying the proposed strategies to price-comparison services.

5. Conclusions and Further Study

The price-comparison service provides the user with an aggregation of item-price information from various shopping malls, but the user still has to navigate a myriad of products

Table 10. The effectiveness with top five similar users

	Ours	user-CF	item-CF	rwr-CF	Popular	Random
1	**0.50**	0.35	0.43	0.29	0.40	0.10
2	**0.76**	0.65	0.55	0.49	0.50	0.40
3	0.14	**0.16**	0.09	0.09	0.00	0.10
4	0.50	0.64	0.71	0.37	**1.00**	0.90
5	**0.50**	0.36	0.21	0.24	0.30	0.20
6	**0.66**	0.50	0.43	0.30	0.30	0.10
7	0.78	**0.81**	0.67	0.38	1.00	0.30
Avg.	**0.55**	0.50	0.44	0.31	0.50	0.30

Table 11. The effectiveness with the most similar user

	Ours	user-CF	item-CF	rwr-CF	Popular	Random
1	**0.60**	0.00	0.43	0.00	0.40	0.10
2	**0.60**	**0.60**	0.50	0.20	0.50	0.40
3	**0.20**	0.00	0.00	0.14	0.00	0.10
4	0.60	0.60	0.75	0.20	**1.00**	0.90
5	**0.90**	0.57	0.17	0.00	0.30	0.20
6	0.50	0.60	**0.75**	0.40	0.30	0.10
7	0.90	**1.00**	**1.00**	0.40	1.00	0.30
Avg	**0.61**	0.48	0.51	0.19	0.50	0.30

to figure out the exact item he is interested in. A personalized recommendation would aid the user to find out what he really wants, which would in turn promote the sales. It is, however, difficult to apply the existing recommendation methods to price-comparison services, because most methods require rating information and suffer from the cold-start and latency problems.

This paper has proposed three recommendation strategies to alleviate the problems with existing methods in price-comparison services. The main contributions of our paper can be summarized as follows:

1. Through our preliminary experiments, we have shown that existing recommendation methods provide quite low accuracy when they applied to click log data.
2. We have identified the characteristics of price-comparison services that cause low recommendation accuracy.
3. In order to increase the recommendation accuracy in price-comparison service sites, we have proposed three recommendation strategies as follows. First, we use click-log data to identify users' preferences. Second, we have similar items grouped together as user's area of interest when capturing users' preferences. Third, by exploiting category hierarchy and keyword information of items, we identify the relationships among user's areas of interest.
4. We have developed a unified framework that reflects the relationships between users and similar-item groups and also between groups in recommendation by using the notion of random walk with restart.

5. We have verified the effectiveness of our framework throughout extensive experiments.

The proposed framework improved maximum of 87% in precision and 129% in recall for regular users, and 184% in precision and 173% in for cold-start users over the existing methods. We have also verified the effectiveness of the recommendation via user studies.

Our proposed framework can be applied to a variety of applications in addition to price-comparison services. They include personalized web page ranking, click-through-based item recommendation, and recommendation in online shopping environment where the problem of one class collaborative filtering may occur. There are several interesting directions for further study. We are currently in the process of incorporating keyword synonyms with feature nodes. By exploiting the notion of synonyms, we expect to increase the data density, thereby achieving higher recommendation accuracy. Also, we are considering to employ the notion of uninteresting items, which are those items unrated but identified as uninteresting to a user, in the one class collaborative filtering problem [20]. We expect it would improve the recommendation accuracy more significantly.

Acknowledgments. This work was supported by (1) the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP; Ministry of Science and ICT) (No. NRF-2017R1A2B3004581) and (2) Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7069440).

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Amer-Yahia, S., Roy, S.B., Chawlat, A., Das, G., Yu, C.: Group recommendation: Semantics and efficiency. Proceedings of the VLDB Endowment 2(1), 754–765 (2009)
3. Baltrunas, L., Makcinskas, T., Ricci, F.: Group recommendations with rank aggregation and collaborative filtering. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 119–126. ACM (2010)
4. Bauer, J., Nanopoulos, A.: Recommender systems based on quantitative implicit customer feedback. Decision Support Systems 68, 77–88 (2014)
5. Berkovsky, S., Freyne, J.: Group-based recipe recommendations: analysis of data aggregation strategies. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 111–118. ACM (2010)
6. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence. pp. 43–52. Morgan Kaufmann (1998)
7. Chae, D.K., Kang, J.S., Kim, S.W., Lee, J.T.: Cfgan: A generic collaborative filtering framework based on generative adversarial networks. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 137–146. ACM (2018)
8. Chae, D.K., Lee, S.C., Lee, S.Y., Kim, S.W.: On identifying k-nearest neighbors in neighborhood models for efficient and effective collaborative filtering. Neurocomputing 278, 134–143 (2018)
9. Chee, S.H.S., Han, J., Wang, K.: Rectree: An efficient collaborative filtering method. In: Proceedings of the 3rd Data Warehousing and Knowledge Discovery, pp. 141–151. Springer (2001)

10. Chen, T., Zhang, W., Lu, Q., Chen, K., Zheng, Z., Yu, Y.: Svdfeature: a toolkit for feature-based collaborative filtering. Journal of Machine Learning Research 13(Dec), 3619–3622 (2012)
11. Cheng, H., Tan, P.N., Sticklen, J., Punch, W.F.: Recommendation via query centered random walk on k-partite graph. In: Proceedings of 7th IEEE International Conference on Data Mining. pp. 457–462. IEEE (2007)
12. Dai, C., Qian, F., Jiang, W., Wang, Z., Wu, Z.: A personalized recommendation system for netease dating site. Proceedings of the VLDB Endowment 7(13) (2014)
13. Fang, X., Pan, R., Cao, G., He, X., Dai, W.: Personalized tag recommendation through nonlinear tensor factorization using gaussian kernel. In: Twenty-Ninth AAAI Conference on Artificial Intelligence (2015)
14. Fu, X., Budzik, J., Hammond, K.J.: Mining navigation history for recommendation. In: Proceedings of the 5th International Conference on Intelligent User Interfaces. pp. 106–112. ACM (2000)
15. Gori, M., Pucci, A.: A random-walk based scoring algorithm with application to recommender systems for large-scale e-commerce. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2006)
16. Gupta, M., Mittal, H., Singla, P., Bagchi, A.: Characterizing comparison shopping behavior: A case study. In: 2014 IEEE 30th International Conference on Data Engineering Workshops. pp. 115–122. IEEE (2014)
17. Hamedani, M.R., Kim, S.W., Kim, D.J.: Simcc: A novel method to consider both content and citations for computing similarity of scientific papers. Information Sciences 334, 273–292 (2016)
18. Han, J., Kamber, M., Pei, J.: Data mining: Concepts and techniques. Morgan Kaufmann (2006)
19. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE International Conference on Data Mining. pp. 263–272. IEEE (2008)
20. Hwang, W.S., Parc, J., Kim, S.W., Lee, J., Lee, D.: "told you i didn't like it": Exploiting uninteresting items for effective collaborative filtering. In: 2016 IEEE 32nd International Conference on Data Engineering. pp. 349–360. IEEE (2016)
21. Hwang, W.S., Lee, H.J., Kim, S.W., Won, Y., Lee, M.s.: Efficient recommendation methods using category experts for a large dataset. Information Fusion 28, 75–82 (2016)
22. Jamali, M., Ester, M.: Trustwalker: a random walk model for combining trust-based and item-based recommendation. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 397–406. ACM (2009)
23. Jamali, M., Ester, M.: Using a trust network to improve top-n recommendation. In: Proceedings of the 3rd ACM Conference on Recommender Systems. pp. 181–188. ACM (2009)
24. Jo, Y.Y., Kim, S.W., Bae, D.H.: Efficient sparse matrix multiplication on gpu for large social network analysis. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. pp. 1261–1270. ACM (2015)
25. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM Journal on Scientific Computing 20(1), 359–392 (1998)
26. Kim, H.N., El-Saddik, A., Jo, G.S.: Collaborative error-reflected models for cold-start recommender systems. Decision Support Systems 51(3), 519–531 (2011)
27. Konstas, I., Stathopoulos, V., Jose, J.M.: On social networks and collaborative recommendation. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 195–202. ACM (2009)
28. Kumar, V., Pujari, A.K., Sahu, S.K., Kagita, V.R., Padmanabhan, V.: Collaborative filtering using multiple binary maximum margin matrix factorizations. Information Sciences 380, 1–11 (2017)
29. Lee, H.J., Kim, J.W., Park, S.J.: Understanding collaborative filtering parameters for personalized recommendations in e-commerce. Electronic Commerce Research 7(3-4), 293–314 (2007)
30. Lee, J., Lee, D., Lee, Y.C., Hwang, W.S., Kim, S.W.: Improving the accuracy of top-n recommendation using a preference model. Information Sciences 348, 290–304 (2016)

31. Lee, S.C., Faloutsos, C., Chae, D.K., Kim, S.W.: Fraud detection in comparison-shopping services: patterns and anomalies in user click behaviors. IEICE TRANSACTIONS on Information and Systems E100-D(10), 2659–2663 (2017)
32. Lee, Y.C., Kim, S.W., Lee, D.: goccf: Graph-theoretic one-class collaborative filtering based on uninteresting items. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
33. Lee, Y., Kim, S.W., Park, S., Xie, X.: How to impute missing ratings?: Claims, solution, and its application to collaborative filtering. In: Proceedings of the 2018 World Wide Web Conference. pp. 783–792 (2018)
34. Leskovec, J.: New directions in recommender systems. In: Proceedings of the Eighth ACM International Conference on Web Search and Data Mining. pp. 3–4. ACM (2015)
35. Li, M., Dias, B.M., Jarman, I., El-Deredy, W., Lisboa, P.J.: Grocery shopping recommendations based on basket-sensitive random walk. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1215–1224. ACM (2009)
36. Li, Y., Hu, J., Zhai, C., Chen, Y.: Improving one-class collaborative filtering by incorporating rich user information. In: Proceedings of the 19th ACM international conference on Information and knowledge management. pp. 959–968. ACM (2010)
37. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing 7(1), 76–80 (2003)
38. Lu, J., Wu, D., Mao, M., Wang, W., Zhang, G.: Recommender system application developments: A survey. Decision Support Systems 74, 12–32 (2015)
39. Oh, H.K., Kim, S.W., Park, S., Zhou, M.: Can you trust online ratings? a mutual reinforcement model for trustworthy online rating systems. Systems, Man, and Cybernetics: Systems, IEEE Transactions on 45(12), 1564–1576 (2015)
40. Onuma, K., Tong, H., Faloutsos, C.: Tangent: a novel,'surprise me', recommendation algorithm. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 657–666. ACM (2009)
41. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Stanford InfoLab (1999)
42. Pan, R., Zhou, Y., Cao, B., Liu, N.N., Lukose, R., Scholz, M., Yang, Q.: One-class collaborative filtering. In: 2008 Eighth IEEE International Conference on Data Mining. pp. 502–511. IEEE (2008)
43. Rafailidis, D., Daras, P.: The tfc model: Tensor factorization and tag clustering for item recommendation in social tagging systems. Systems, Man, and Cybernetics: Systems, IEEE Transactions on 43(3), 673–688 (2013)
44. Sandvig, J.J., Mobasher, B., Burke, R.: Robustness of collaborative recommendation based on association rule mining. In: Proceedings of the 1st ACM conference on Recommender Systems. pp. 105–112. ACM (2007)
45. 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. ACM (2001)
46. Shapira, B., Ricci, F., Kantor, P.B., Rokach, L.: Recommender systems handbook. (2011)
47. Shi, Y., Larson, M., Hanjalic, A.: Exploiting user similarity based on rated-item pools for improved user-based collaborative filtering. In: Proceedings of the 3rd ACM Conference on Recommender Systems. pp. 125–132. ACM (2009)
48. Tong, H., et al.: Fast random walk with restart and its applications. In: Proceedings on 6th International Conference on Data Mining. pp. 613–622 (2006)
49. Tran, T., Lee, K., Liao, Y., Lee, D.: Regularizing matrix factorization with user and item embeddings for recommendation. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. pp. 687–696. ACM (2018)
50. Wan, M., McAuley, J.: One-class recommendation with asymmetric textual feedback. In: Proceedings of the 2018 SIAM International Conference on Data Mining. pp. 648–656. SIAM (2018)

51. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 448–456. ACM (2011)
52. Wang, F., Ma, S., Yang, L., Li, T.: Recommendation on item graphs. In: Proceedings on 6th International Conference on Data Mining. pp. 1119–1123. IEEE (2006)
53. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 1235–1244. ACM (2015)
54. Yang, L., Cui, Y., Xuan, Y., Wang, C., Belongie, S., Estrin, D.: Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In: Proceedings of the 12th ACM Conference on Recommender Systems. pp. 279–287. ACM (2018)
55. Yildirim, H., Krishnamoorthy, M.S.: A random walk method for alleviating the sparsity problem in collaborative filtering. In: Proceedings of the 2nd ACM Conference on Recommender Systems. pp. 131–138. ACM (2008)
56. Yoon, S.H., Kim, S.W., Park, S.: C-rank: A link-based similarity measure for scientific literature databases. Information Sciences 326, 25–40 (2016)
57. Yu, X., Ma, H., Hsu, B.J.P., Han, J.: On building entity recommender systems using user click log and freebase knowledge. In: Proceedings of the 7th ACM international conference on Web search and data mining. pp. 263–272. ACM (2014)

Sang-Chul Lee received the B.S., M.S., and Ph.D. degrees in Electronics and Computer Engineering from Hanyang University, Seoul, Korea, in 2005, 2007, and 2012, respec tively. He worked as a postdoctoral researcher at Carnegie Mellon University from 2013 to 2015. Currently, he is a senior engineer at Hyundai Heavy Industries. His research interests include data mining, information retrieval, and recommender systems.

Sang-Wook Kim received the B.S. degree in computer engineering from Seoul National University, in 1989, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), in 1991 and 1994, respectively. From 1995 to 2003, he served as an associate professor at Kangwon National University. In 2003, he joined Hanyang University, Seoul, Korea, where he currently is a professor at the Department of Computer Science and Engineering and the director of the Brain-Korea-21-Plus research program. He is also leading a National Research Lab (NRL) Project funded by the National Research Foundation since 2015. From 2009 to 2010, he visited the Computer Science Department, Carnegie Mellon University, as a visiting professor. From 1999 to 2000, he worked with the IBM T. J. Watson Research Center, USA, as a postdoc. He also visited the Computer Science Department at Stanford University as a visiting researcher in 1991. He is an author of more than 200 papers in refereed international journals and international conference proceedings. His research interests include databases, data mining, multimedia information retrieval, social network analysis, recommendation, and web data analysis. He is a member of the ACM and the IEEE.

Sunju Park is a professor of operations, decisions and information at the School of Business at Yonsei University, Seoul. Her education includes a B.S. and M.S. degrees in computer engineering from Seoul National University and a Ph.D. degree in computer science and engineering from the University of Michigan. Before joining Yonsei University, she has served on the faculties of management science and information systems at Rutgers

University. Her research interests include analysis of online social networks, multiagent systems for online businesses, and pricing of network resources. Her publications have appeared in Computers and Industrial Engineering, Electronic Commerce Research, Transportation Research, IIE Transactions, European Journal of Operational Research, Journal of Artificial Intelligence Research, Interfaces, Autonomous Agents and MultiAgent Systems, and other leading journals.

Dong-Kyu Chae received the B.S. and Ph.D. degrees in computer science from Hanyang University, Seoul, Korea, in 2012 and 2019, respectively. From June 2019, he will visit the School of Computational Science & Engineering at Georgia Institute of Technology as a visiting researcher. His current research interests include data mining, social network analysis, recommender systems and deep learning.

Received: October 12, 2018; Accepted: April 2, 2019.

Computer Science and Information Systems 16(2):359–380 https://doi.org/10.2298/CSIS180703006P

Product Reputation Mining: Bring Informative Review Summaries to Producers and Consumers

Zhehua Piao[1], Sang-Min Park[2], Byung-Won On[2], Gyu Sang Choi[3], and Myong-Soon Park[1]

[1] Department of Computer Science and Engineering, Korea University,
145 Anam-ro, Seongbuk-gu, Seoul 02841, South Korea
huanmie2199@gmail.com, myongsp@korea.ac.kr
[2] Department of Software Convergence Engineering, Kunsan National University,
558 Daehak-ro, Gunsan-si, Jeollabuk-do 54150, South Korea
{b1162, bwon}@kunsan.ac.kr
[3] Department of Information and Communication Engineering, Yeungnam University,
280 Daehak-ro, Gyeongsan-si, Gyeongbuk-do 38541, South Korea
castchoi@ynu.ac.kr

Abstract. Product reputation mining systems can help customers make their buying decision about a product of interest. In addition, it will be helpful to investigate the preferences of recently released products made by enterprises. Unlike the conventional manual survey, it will give us quick survey results on a low cost budget. In this article, we propose a novel product reputation mining approach based on three dimensional points of view that are word, sentence, and aspect–levels. Given a target product, the aspect–level method assigns the sentences of a review document to the desired aspects. The sentence–level method is a graph-based model for quantifying the importance of sentences. The word–level method computes both importance and sentiment orientation of words. Aggregating these scores, the proposed approach measures the reputation tendency and preferred intensity and selects top-k informative review documents about the product. To validate the proposed method, we experimented with review documents relevant with K5 in Kia motors. Our experimental results show that our method is more helpful than the existing lexicon–based approach in the empirical and statistical studies.

Keywords: product reputation mining, opinion mining, sentiment analysis, sentiment lexicon construction

1. Introduction

Data analysis strategies and technologies are widely used in the recent marketing research area. In order to investigate the preferences of recently released products, enterprises need a state–of–the–art strategy to accurately grasp the public's taste for the product by automatically collecting and analyzing various types of data on the Web. In the manual survey, if company executives want to know how consumers think of their brand–new product, the employees in the marketing department will conduct a survey via email and phone. However, the recent survey response rate is low because modern people do not have enough time to response sincerely to the questionnaire and recent consumers are mainly interested in customized products. Unlike this conventional process, product reputation systems that

collect various online data and predict the accurate survey results can provide quick results on a low cost budget. Meanwhile, potential customers may save their time in making their buying decision if they are served by product reputation mining systems. Until now, to purchase a brand-new product like Hyundai Sonata, a customer is likely to spend a lot of time in gathering helpful information on the Web. He/she first attempts to search for Hyundai Sonata and carefully go over relevant web pages one by one. Even though he/she strives to figure out which model is better, it is difficult to take the clear point due to a lot of information and advertising. As above, the product reputation mining systems can provide many benefits to both producers and consumers.

In this section, we briefly define the product reputation mining system as:

- In the first problem, given a product of interest, it automatically measures the reputation tendency (sentiment orientation – positive or negative) and level (the intensity of the sentiment orientation) of various aspects (e.g., price, design, and service) of the product. We assume that all aspects of the target product are given in advance.
- In the second problem, for each aspect of the product, it selects the top–k documents including the most *informative* reviews in the corpus of review documents. We assume that all review documents irrelevant with the target product are already filtered out before this problem. In addition, we will carefully define how informative a review document is in the next section.

Through the proposed product reputation mining system, both companies and customers can easily know the public's preference (as positive or negative) and the preferred intensity (Level $1 \sim 5$) of a product. They can also know the detailed points with respect to each aspect of the product. For the details, please see Section 3.

To address the product reputation mining problem, in this work, we propose a novel three-dimensional reputation mining approach that consists of aspect, sentence, and word–level methods.

As shown in Figure 1, the aspect–level method is the aspect classification model based on SVM, Random Forest, and FNN to assign the sentences of review documents to the desired aspects. The sentence–level method is a graph-based model for quantifying the importance of each sentence in review documents. The word–level method computes the importance of a word and measures the sentiment score of the word based on Korean sentiment lexicon. Finally, aggregating the scores of the aspect, sentence, and word–level methods, our method measures the reputation tendency and level in each aspect of the target product. In addition, all review documents in each aspect are rearranged by the aggregated scores and then top-k review documents with the highest aggregated scores are selected as the informative documents.

Our experimental results show that the accuracy of the aspect–level method is at least 0.852. In the existing lexicon–based approach, F_1–scores of the positive and negative sentences are 0.678 and 0.688, while those are 0.758 and 0.795 in the proposed method. These results mean that the proposed method improves about 12% and 5% in the positive and negative sentences. In our case study for K5 in Kia motors, we observed top-k reviews retrieved by the proposed method and finally concluded that most of the review documents are informative. We will discuss the experimental results In Section 4 and conducted a user study for the results retrieved by the proposed method and performed statistical tests. Through the significance tests, it turns out that our method is statistically better than the existing method.

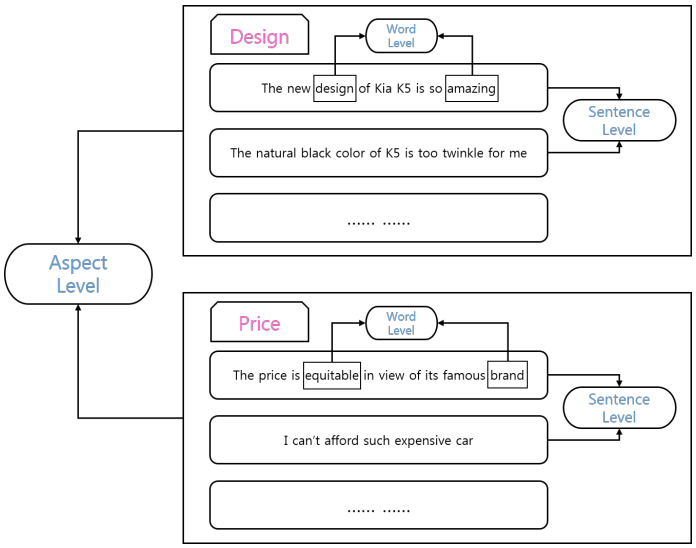


Fig. 1. Aspect, Sentence, and Word–level based product reputation mining approach

The contributions of our work are as follows:

- To address the product reputation mining problem, we propose a novel three–dimensional reputation mining approach that consists of aspect, sentence, and word–level methods. We show the detailed algorithms of (1) measuring the reputation tendency and level about a target product and (2) selecting top-k informative review documents. These results will help customers make their buying decision and companies get to know the public's preference about the product in detail. We also constructed an elaborate Korean sentiment lexicon to determine the sentiment orientation of words.
- Our experimental results show that the proposed method is effective to address the product reputation mining problem. Compared to the existing lexicon–based approach, it improves up to 12% F_1– score. In addition, our statistical verification shows that the proposed method will be helpful for both company employees and customers. Consequently, these results indicate that it is beneficial to develop a web–based system based on the proposed method of aggregating three dimensional sentiment scores.
- According to our intensive literature survey, the key point of our method is to quantify the reputation of the product based on three dimensions (i.e., aspect, sentence, and word–levels). To the best of our knowledge, this is the first study to tackle the product reputation mining problem.

The remainder of this article is organized as follows: In section 2, we introduce previous main machine learning and lexicon–based approaches related to this work. In particular, we discuss the novelty of our method, in addition to the difference between previous studies and our work. In section 3, we deal with the formal problem definition. Then, we describe our product reputation mining approach in detail in section 4. Next, we explain the experimental set-up and discuss the experimental results in Section 5. Finally, we conclude our work and mention the future research direction in Section 6.

2. Literature Review

In 2000, Resnick et al. presetned several challenging issues and solution overview of product reputation systems that collect, distribute, and aggregate feedback about past consumers' behaviour so that these systems help people make their buying decision based on public history of particular sellers. By showing real cases of eBay's auction site, Bizrate's survey forum, and iExchange's product review site, the authors also stated main requirements of the product reputation systems. In particular, they focused on gathering reliable feedback in the reputation systems. For the detail, please see [18].

In 2008, Hwang and Ko proposed a Korean sentiment analysis method of labelling a document to either positive or negative and of classifying a sentence to either subjective or objective [9]. In the method, the authors made a Korean sentiment lexicon in which a Korean word is translated to the corresponding English word to obtain the polarity of the word. In addition, the sentiment analysis method classifies documents and sentences based on Support Vector Machines (SVM). However, their method does not consider aspects that are recently considered to be important in the product reputation mining problem.

Given a particular product, Jin and Ho presented a machine learning technique based on lexicalized Hidden Markov Models (HMMs) [10]. Specifically, their method extracts subjective sentences related to the target product from review documents and then labels each sentence to either positive or negative class. In particular, they trained the lexicalized HMMs with linguistic features including part-of-speech, phrases' internal formation patterns, and contextual clues surrounding words and phrases. Applying such linguistic features to HMM is different from previous approaches that address the product reputation mining problem. However, there is still room to improve the accuracy of the existing methods and to identify more informative review documents than the entire documents. Our proposed method shows the better result of correctly classifying the sentiment orientation of a target product based on aggregating the reputation scores measured by aspect, sentence, and word–level algorithms. For the detail, please refer to the experimental result section.

Steinberger et al. proposed a new approach that semi-automatically creates sentiment dictionaries in several languages [21]. They first made sentiment dictionaries for two source languages and then automatically translated them to the third languages in which a word is likely to be similar to that of the source languages. Through the third languages, the target dictionaries can be more corrected and further extended. In the experiment, they validated such a triangulation hypothesis by comparing triangulated lists to non–triangulated machine–translated word lists.

Khose and Dakhode proposed a product reputation analysis system that consists of five steps [12]. In the first step, review documents are collected and pre-processed for the next step. After target and opinion words are extracted, an object relation graph is formed by detecting the relation between them. In the third step, the weight of a node called confidence is computed based on a simple random walk model. In addition, to determine the sentiment scores of the target word and its opinion words, they used SentiWordNet that is a popular sentiment lexicon in opinion mining area. Each target word is represented as a vector of the target word's confidence and the sentiment scores of the opinion words associated with the target word. The reputation score of the target word is finally calcu-

lated as the product of the summation of the vectors related to the target word. Unlike our method, they consider only the reputation score of a target product in word–level.

In 2016, rather than classifying the porality (i.e., positive or negative) of words, Canales et al proposed a bootstrapping method that labels an emotional corpus automatically [4]. Based on NRC Word–Emotion Association Lexicon, they created the seed set and then expended the initial seed by means of similarity metrics. Furthermore, to distinguish between emotion categories in a fine–grained lexicon with 28 emotion categories, the authors in [26] proposed an approach of labelling primary and secondary words to one of emotion categories, where the primary words are used for detecting synonyms or other semantic words associated with each category, while the secondary words are used to mine the contextual relation between words. However, these methods focus mainly on constructing a fine-grained emotion lexicon which is considerably different from the production reputation mining problem that we present in this article. In addition, Ko presented a general method for creating an emotional word dictionary containing a semantic weight matrix and a semantic classification matrix [13]. Based on clustering synonymous relations and frequencies, he showed the detailed process of collecting a classification and weight matrix that can be used as the ontology and linked data of emotion.

Meanwhile, detecting emotion from text documents is a non-trivial task because of the limitation of human annotation. To tackle this problem, the authors in [25] utilized emoji as self–annotation of twitter users' emotional status. They believed that emoji is a good emotion indicator presenting a faithful representation of a user's emotional status but their approach is too limited to use other text documents rather than tweet mentions.

Sentiment analysis is generally categorized to two groups. One is machine learning approach and the other is lexicon-based approach. Although the lexicon-based approach has been used in wide applications, it does not work well to determine the sentiment orientation of tweets. This is because each tweet document is limited to only 140 characters and its sentences are not written according to the grammar. [19] presented SentiCircles, a lexicon-based approach for Twitters, that is based on the co-occurrence patterns of words in different tweet documents to update the prior degree and polarity in sentiment lexicons accordingly.

To improve the accuracy of the machine learning approach, Long Short Term Memory (LSTM) as one of Recurrent Neural Networks (RNN) deep learning models is widely used to classify a text document to either positive or negative class. LSTM is trained with a large number of training set containing a large number of pairs of the text document and sentiment polarity manually annotated by human experts. Then, given a text document in the test set, LSTM automatically determines the sentiment orientation of the text document. Teng et al. proposed a hybrid approach that is the trade-off between the context-sensitive method using LSTM and the lexicon-based method using the list of sentiment words [22]. This approach is not one of the product reputation mining algorithms but an advanced method for improving the conventional sentiment document classification methods. Similarly, [23] presented a hybrid approach that measures numerical numbers in multiple dimensions (i.e., valence-arousal space) by extracting/abstracting the locality information within each sentence based on Convolutional Neural Networks (CNN) and updating the context weights by means of long–distance dependency cross sentences based on LSTM.

Nowadays, online travel forums and social network sites are popular for sharing travel information. Review summaries for hotels automatically generated from many reviews in the sites can help travelers choose their preferred hotel during the trip. For (opinion) mining from online review documents about a target hotel, Hu et al. proposed a summarization method that finds top–k sentences using k–medoids clustering algorithm that removes sentences irrelevant with the target hotel [8]. They also proposed additional feature set that includes author reliability, review time, review usefulness, and conflicting opinions which are not considered in the previous review summarization methods. Although Hu et al.'s method is similar to our proposed method in that it selects top–k relevant sentences from review documents, there are main differences between them. While Hu's method first clusters text documents by contexutal information and then selects only top–k relevant sentences, our method computes the reputation score of a target product using word, sentence, and aspect–level methods and identifies top–k relevant but yet informative sentences. In addition, we make use of various learning models such as SVM, Random Forest, and even deep neural networks, whereas Hu's method is based on only k–medoids clustering method.

3. Problem definition

Table 1. An example of the first solution method

Aspect	Reputation tendency	Reputation level
Design	Positiveness	Level 2
Performance	Positiveness	Level 3
Price	Negativeness	Level 1
Quality	Positiveness	Level 5
Service	Positiveness	Level 5

In this section, we define the *product reputation mining* problem as two sub-problems. In the first problem, given a product of interest as input (e.g., a particular car e like K5 made by Hyundai and KIA motors), the goal of the product reputation mining method is to *automatically* measure the reputation tendency and level per aspect. For instance, *design*, *performance*, *price*, *quality*, and *service* may be the main aspects that many consumers often consider importantly when they are about to purchase their brand-new car. Table 1 shows the outcome of the product reputation mining method. Let us assume that design, performance, price, quality, and service are given in advance as the main aspects of evaluating general vehicles. Actually several domain experts recommended the five aspects to us. For each aspect, the product reputation mining method will label the reputation tendency to either positiveness or negativeness. The reputation level indicates the intensity of the reputation tendency. In our context, there are five levels in positiveness, neutrality, and five levels in negativeness. The five levels are specified to Level 1 $\sim$ Level 5. The strongest positiveness (negativeness) is Level 5, while the weakest positiveness (negativeness) is Level 1.

Next, to tackle the second problem, the product reputation mining method finds top–k documents including both *relevant* and *informative* reviews in the corpus. The top–k documents seldom contain meaningless advertisement and exaggeration, spam/fake reviews, and even text content which is not directly related to e. Here are two examples that we collected in the most popular web site with many reviews regarding vehicles in Korea. The following document is considered to be both relevant and informative.

> Title: Kia's next-generation K5 does not change but actually changed everything
> Author: Charisma4097
>
> The interior was completely obscured and difficult to identify, but the overall shape could be guessed. Once the change is expected to be quite large. Unlike the existing design that surrounds the driver's seat, it is likely to change to a horizontal feeling that stretches from the driver's seat to the next seat. The door design is completely different from that of the existing K5. Steeply raised buttons and knobs are flat. Sheets are very similar to those in the new Sonata. It is characterized by large and wide, with the middle vertical line. However, I am not sure that I will use this sheet similar to the new Sonata.

On the other hand, the following document shows the typical document that does not help consumers at all.

> Title: Someone who takes the new K5, What about the breaks?
> Author: Mr. Oral
>
> While I am getting ready to change from SM5 to K5, I wonder if there are too much talk about the bad breaks. I also wonder what K5 Turbo JBL sound is like. Once I heard from Mark Levinson audio in Lexus, it was so cool.

In practice, the sentiment analysis is the most important step in the product reputation mining problem. The sentiment analysis is generally categorized to two approaches [16]. One is the machine learning approach and the other is the lexicon–based approach that is also divided to dictionary–based and corpus–based approaches. Nowadays, even though main deep learning models such as CNN and RNN used for sentiment analysis have shown better results, the lexicon–based approach is still important. In a particular domain, the accuracy of the lexicon–based approach is much higher than machine learning approaches. Thus, sentiment analysis based on sentiment lexicons has been widely used in practical applications. In addition, the lexicon–based approach has no need for complex environment setting like GPU or long pre–training time before the learning model is used [1]. More importantly, for greater accuracy, the existing deep learning models need large–scale training data set. In fact, it is non–trivial to obtain the large–scale training data because human annotators need to label the classes manually. To avoid this problem, state–of–the–art researches are being carried out to pseudo–generate the large–scale training data using sentiment lexicons. Due to these reasons, the lexicon–based approach is still the important methodology in sentiment analysis. In this study, we focus only on the improvement of the existing lexicon–based approaches [7].

4. Main Proposal

To address the product reputation mining problem, we first consider the three dimensional coordinate system in which x-axis, y-axis, and z-axis indicate the aspect matching score, sentence importance score, and word sentiment score of a product of interest, respectively. In our problem, given a particular product, the three various scores are first measured by our proposed aspect classification method, sentence weight estimation method, and word sentiment scoring method, and then are aggregated to its total sentiment score.

Now we briefly summarize the key concept of the proposed three methods – the aspect classification method, sentence weight estimation method, and word sentiment scoring method. We will also describe the detailed algorithms in the following subsections. In our aspect classification method, we assume that five aspects of cars such as design, performance, price, quality, and service are given in advance. The aspects were manually decided by several experts in the automobile domain. Given a review document as input, it is divided to a set of sentences. Each sentence is automatically classified to one of the five aspects.

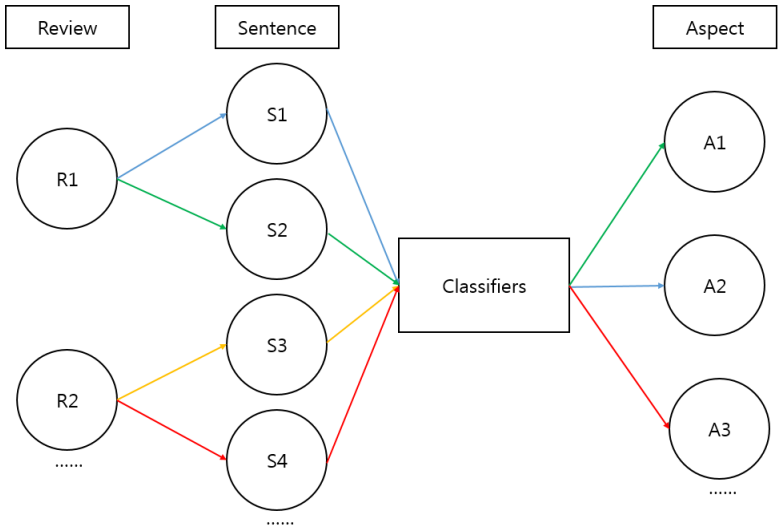


Fig. 2. Diagram of our aspect matching method

Figure 2 shows the diagram of the proposed aspect matching method and the detailed algorithm is depicted in Algorithm 1. The train set is a set of pairs like (sentence, one of five aspects (i.e., price, performance, design, service, and quality)) that is stored as a list of nodes, each of which contains a sentence and an aspect, in the main and secondary storage. To train a learning model and to conduct the test step, we use Support Vector Machine (SVM)[5], Random Forest[3], and Feed-forward Neural Network (FNN)[15]. SVM was the best classification method before deep learning models are employed actively. Random Forest often provides high accuracy because it is the best ensemble method. Unlike the conventional classification methods, it is known that FNN works effectively because of the deep neural network with multi hidden layers when we attempt to cope with the non-linear classification problem. Since each model has its pros and cons, the three learning models are used to assign sentences to the desired aspects.

Subsequently, to identify important sentences in a review document, we propose a graph-based model for estimating sentence weight values. After the review document is segmented to a set of sentences, each sentence is represented as a vertex in a graph G. The link weight between two nodes n_1 and n_2 is the similarity value $sim()$ between the

Algorithm 1 Aspect Matching Method

Require: The whole review data set about a given product DS;
Ensure: Several sub corpus with the corresponding to the labels;

```
Use classifiers to classify sentences in DS:
if sentences ∈ classifier 1 then
    Align sentences to sub corpus about label 1;
else
    Send to classifier about label 2 to be classified;
    if sentences ∈ classifier 2 then
        Align sentences to sub corpus about label 2;
    else
        Send to classifier about label 3;
                        .....
        if sentences ∈ classifier n then
            Align sentences to sub corpus about label n;
        else
            These sentences are irrelevant;
        end if
        .....
    end if
end if
```

sentences corresponding to n_1 and n_2. If $sim(n_1, n_2) < \theta$ (a certain threshold value), the link between n_1 and n_2 is removed in G. To measure the connection strength between n_1 and n_2, our method is to compute the probability value to reach n_2 from n_1 via random walks over G, where for each step, random walkers visit neighbouring nodes with a certain probability. This graph-based method is based on the underlying assumption that more important sentences are likely to receive more links from other sentences. We will discuss the similarity and graph–based probability equations in Section 4.1.

Next, in word–level, we focus on both importance and sentiment orientation of words in a review document. To quantify the importance of a word, we use Term Frequency / Inverse Document Frequency (TF/IDF) that is widely used in the information retrieval community. Through TF/IDF metric, a word w is considered to be important if w appears many times in a document, while w seldom appears in the entire corpus. To compute the sentiment degree value of w, we constructed a sentiment lexicon for Korean language with assistance from Korean linguists. As illustrated in Figure 3, the Korean sentiment lexicon consists of the list of positive and negative words, incrementer and decrementer, flip words, and conjunction words. Based on the sentiment lexicon, we propose a word–level sentiment scoring method that computes the final score by merging the TF/IDF and sentiment scores of w. For the detail, we will discuss the detailed algorithm in Section 4.1.

Finally, after the above three methods are performed, each sentence is assigned to (`word--level score, sentence--importance score`) called *sentence reputation score (υ)*. For each aspect a, the total score υ_p of all positive sentences related to a are calculated. In the same way, the total score υ_n of all negative sentences related to a are calculated as well. Then, the reputation tendency and level are approximated based on υ_p and υ_n. Furthermore, the document reputation score υ_d is computed by $\Sigma_{i=1}^{k} \upsilon_i$, where k

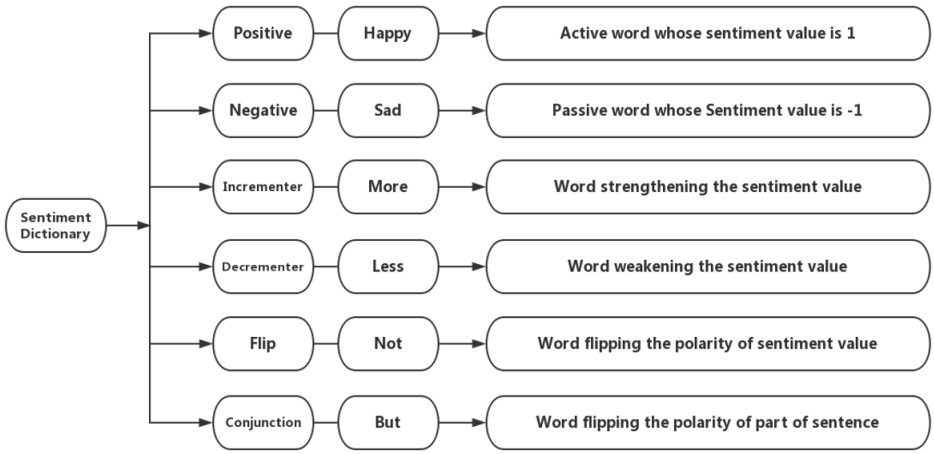


Fig. 3. A Korean sentiment lexicon

is the number of sentences in the document. In the final step, all review documents in the corpus are rearranged by v_d and then top-k review documents are chosen as informative ones.

Suppose that n is the number of sentences in the collection of reviews as input. Because we propose a FNN-based aspect matching model and compare it to the existing learning models such as SVM and Random Forest, we focus merely on computing the time complexity of FNN. Please refer to Table 3 in the paper. According to the table, each word of the sentence is converted to a 100-dimensional word embedding vector. If each sentence is composed of ten words, the dimension number of the input vector is 1,000. If the number of words is below ten, we put zero values to empty dimensions (as a padding approach). We develop the FNN model with five hidden layers (H1, ..., H5) that contain 1,000, 800, 600, 400, and 200 units, respectively. The input layers has 1,000 units and the output layer has 5 units (# of aspects). The number of the weight parameters between the input layer and H1 is 1,000∗1,000 and the number of biases between them is 1,000. Similarly, the number of the weight parameters between H1 and H2 is 1,000∗800 and the number of biases between them is 800. As a result, the total number of parameters is (1,000∗1,000+1,000) + (1,000∗800+800) + (800∗600+600) + (600∗400+400) + (400∗200+200) + (200∗5+5) = 2,604,005. This means that at least 2,604,005 memory spaces are required in both train and test sets. Since FNN model finds optimal parameters through forward and backward propagation, the time complexity is dominant to the number of computing the parameters between the input layer and H1. In other words, in case of the number of units in the input layer is n, it takes $O(n * n + n) = O(n^2)$.

4.1. Aspect Classification Models

Sentence Importance Estimation Method In this section, we present the similarity and graph-based probability equations by which the importance of each sentence in review documents is quantified. The similarity equation (sim) between two sentences s_i and s_j is defined as:

$$sim(s_i, s_j) = \frac{|\{w_k | w_k \in s_i, w_k \in s_j\}|}{log(|s_i|) + log(|s_j|)} \quad (1)$$

Eq. (18) means that the numerator is the number of the overlapped words between two sentences and the denominator is the length of the two sentences mainly used to normalize the similarity score. This proposed similarity measure is reasonable to see how similar the two sentences are and the meaning of the proposed similarity method is close to Jaccard similarity measure that is simple but yet high-accurate so is widely used in real applications. In Eq. (18), $|s_i|$ and $|s_j|$ are the numbers of words in s_i and s_j and the numerator indicates the number of the words appearing in both s_i and s_j.

To measure the importance of each node in a graph, where a node stands for a sentence, we refer to as:

$$\upsilon_i = (1-\gamma)\frac{1}{n} + \gamma \Sigma_{j \in degree(i)} \frac{sim(i,j)}{\Sigma_{k \in degree(i) \wedge k \neq j} sim(i,k)} \upsilon_j \quad (2)$$

Eq. (19) is proposed to identify the global sentiment score of each sentence. It works based on random walks, where the local sentiment score of a sentence (e.g., x) is propagated to neighbor sentences (e.g., y and z) with its probability values (similarity between x and y, similarity between x and z). For example, suppose that xs sentiment score is 0.9; the weight between x and y is 0.7; and the weight between x and z is 0.3. In this case, a random walker visits to y from x at a probability of 0.7, while it also visits to z from x at a probability of 0.3. We believe that the equation makes sense to find each sentences optimal score by considering both the importance and sentiment of all sentences in the collection. In Eq. (19), i, j, and k are nodes and $degree(i)$ means a set of the neighbouring nodes of i. n is # of nodes in the graph and γ is the weight value of each equation term. Starting at node i, random walks continue to visit the neighbouring nodes until they arrive at all nodes in the graph to compute the probability value of i which is denoted by $P(i)$. The bigger $P(i)$ is, the higher the importance of i is. This is, if one node is pointed by important nodes in the graph, it may also be an important node. In Eq. (19), the first term $\frac{1}{n}$ needs because a random walker jumps to another node chosen at random with the equal probability whenever it meets terminal nodes in the graph. To quantify the weight of each sentence in a given corpus, the similarity between two sentences is computed and stored as a square matrix, where each row(column) means a sentence. In addition, the sentiment scores of all sentences are stored in a vector. The matrix-vector multiplication is performed iteratively until the values of the vector are converged. Suppose that n is the number of sentences in the collection of reviews as input. A n by n matrix and a n-dimensional vector are created, where the matrix contains the similarity values between two sentences and the vector means the sentiment score of each sentence. The space complexity is $O(n^2 + n) = O(n^2)$. The algorithm is peformed iteratively until there in no difference between the previous and current values in the vector. If we consider k to be the average number of iterations, the algorithm does the matrix-vector multiplication by k times. In each matrix-vector multiplication, the total number of the multiplications is n*n and the total number of the additions is n-1. For n rows in the matrix, the multiplications and additions are needed so $O(n(n * n + (n-1))) = O(n^3)$. As a result, the time complexity is $O(k * n^3)$. Algorithm 2 shows the detailed algorithm of quantifying the importance of sentences.

Algorithm 2 Sentence–level Method

Require: One sub corpus processed by Algorithm 2: C;
Ensure: Reputation score for each sentence: srs;

1: Each sentence in the corpus gains a tr by Eq. (19);
2: **for** sentence in C **do**
3: srs = $ssc * tr$;
4: **end for**

Word Sentiment Scoring Method This method consists of two terms of the equation. For each word w, one is to measure the importance of w and the other is to measure the sentiment score of w. To quantify the importance of w, we use TF/IDF metric, where TF is Term Frequency meaning the frequency of a word within a document. For example, if a word 'obama' appears three times in a document that has 100 words, TF('obama')= $\frac{3}{100}$ = 0.03. On the other hand, IDF is Inverse Document Frequency, indicating that a word is more important if it is unique in the corpus. Suppose that we have 10 million documents in the corpus. If 'obama' appears in only 1,000 documents, then IDF('obama')=$\frac{10{,}000{,}000}{1{,}000}$=4. As a result, TF/IDF(obama)=0.03 $\times$ 4=0.12. In this way, the weight value of each word is quantitatively computed using TF/IDF which is between 0 and 1. If the weight value of the word is close to 1, then it means that the word is very important. On the other hand, if the weight value is low, the corresponding word is trivial. Such a word may be 'a', 'the', 'in', and so on. This weight value of each word captures the importance of the word. Meanwhile, to compute the sentiment

Algorithm 3 Word–level Method

Require: One sub corpus after aspect classification: C;
Ensure: Reputation score for each sentence: ssc;

1: Each word in the corpus gains a ti value by TF/IDF;
2: **for** sentence in C **do**
3: **def** sentence_rs(sentence_tokens, pw, nw, ssc):
4: **if not** sentence_tokens **then**
5: **return** ssc;
6: **else**
7: cw = sentence_tokens[0];
8: Gain the wss of cw from Rules;
9: ssc = ssc + $wss * ti$;
10: **if** nw $\in$ Conjunction dictionary **then**
11: ssc = $ssc * (-1)$;
12: **end if**
13: **return** sentence_rs(sentence_token[1:], cw, nw, ssc)
14: **end if**
15: **end for**

score of w, we constructed and used our own Korean sentiment lexicon as shown in Fig-

ure 3. In particular, the sentiment dictionary contains positive and negative words, incrementer/decrementer, flip words, and conjunction words. The final word–level score(w) = TF/IDF(w) $\times$ sentiment–score(w). Each sentence is tokenized to words and then is stored as a list of pairs like (Sentence ID, [$word_1$, $word_2$, $word_3$, ...]). In addition, HashMap is used, where the key is (Sentence ID, $word_i$) and the value is (TF/IDF and sentiment scores of $word_i$). Suppose that n is the number of sentences in the collection of reviews and each sentence contains m words on average. In the first step, all TF/IDF and sentiment scores are computed by n$*$m times. In the second step, Algorithm 3 is performed by n$*$m times. Thus, the time complexity is $O(n * m)$. Meanwhile, to store a list that contains the pairs of (Sentence ID, Words), it needs n$*$m+n spaces, where n$*$m means the total number of words in sentences and n means the number of the sentences identifiers. We also need a HashMap, where each key needs 2 for storing a sentence ID and each word, and each value needs 2 for storing TF/IDF and sentiment scores. Thus, the HashMap needs $O(n * m(2 + 2)) = O(n * m)$ spaces. As a result, the total space complexity is $O((n * m + n) + (n * m)) = O(n * m)$. Algorithm 3 describes the detailed procedure.

Estimation of Reputation Tendency and Level As the final result, each sentence is labelled to sentence reputation score (v)=(`word--level score, sentence--importance score`). For each aspect a, the total score v_p of all positive sentences related to a are calculated and then the final reputation score is estimated based on $\frac{v_p}{|v_p|+|v_n|}$. In the same way, the total score v_n of all positive sentences related to a are also calculated and then the final reputation score is estimated based on $\frac{v_n}{|v_p|+|v_n|}$. Finally, the reputation scores are transformed to the relevant reputation tendency and level based on the index table in Figure 4.

PRR	0≤	10%≤	20%≤	30%≤	40%≤	50%≤
Reputation Level	Neg Lv.5	Neg Lv.4	Neg Lv.3	Neg Lv.2	Neg Lv.1	Neutral
PRR	60%≥	70%≥	80%≥	90%≥	100%≥	
Reputation Level	Pos Lv.1	Pos Lv.2	Pos Lv.3	Pos Lv.4	Pos Lv.5	

Fig. 4. Reputation tendency and level index

To find informative review documents, the document reputation score v_d is computed by $\Sigma_{i=1}^{k} v_i$, where k is the number of sentences in the document. In the final step, all review documents in the corpus are rearranged by v_d and then top-k review documents are chosen as the informative documents.

5. Experimental Validation

5.1. Experimental Set–up

In the previous section, we described the detailed algorithms of the proposed approach for computing the reputation tendency and level and selecting top–k informative sen-

tences about a target product. Now we introduce the process of evaluating the proposed method, comparing to a straightforward lexicon–based approach as the baseline method with online reviews about K5 in Kia motors. We collected 1,585 review documents in Bobaedream, the most popular web sties related to car reviews. In the pre–processing step, we replaced all words by lower–case letters after removing images, moving pictures, and advertising texts. Then, we removed stop words [24] in the all documents and converted derived words to root forms through a stemming software [17]. After the pre–processing step, we collected 1,562 review sentences. To make the gold standard set (solution set), four human annotators subjectively labelled the aspect of each sentence to one of five aspects (design, performance, price, quality, and service) and conflicting sentences are decided by a majority vote. In the same way, they manually classified all sentences to a particular sentiment orientation (positive, neutral, and negative). For example, given a sentence "The front design with Raff is very good," the sentence orientation is positive and the aspect label is design. Figure 5 shows the brief characteristics of the data set.

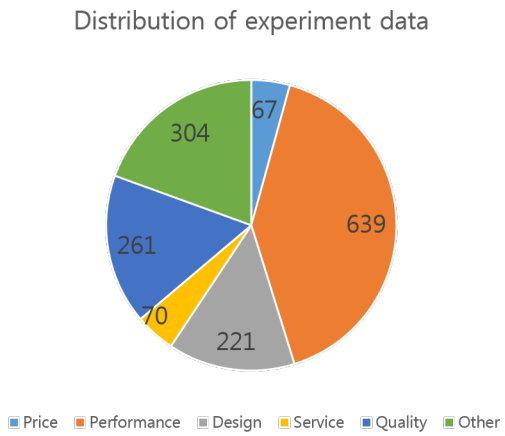


Fig. 5. Distribution of the review documents across five aspects

To select the discriminative features of input vectors, we first computed TF/IDF values of all words in the data set, and then used top–k words with the highest TF/IDF values as the feature set. For example, # of the words in the feature set is 1,000. In our repetitive experiments, we carefully investigated the results of all methods for all possible cases to find the optimal number of the features in the data set. Finally, after making feature vectors based on the feature set in the data set, we converted the feature vectors to the input vectors, which is the input of the models used in our experiment, using a popular word embedding method such as Word2Vec [20].

We implemented the aspect matching method based on FNN deep learning model in Python and TensorFlow [6]. The experimental set–up of the method used in our experiments is summarized in Table 2. Through our intensive experiment, we found the optimal values of the hyper parameters that are suitable in our problem. For the initial values of weight parameters, we used the truncated normal method [14]. As an activation function,

ReLU was used in the entire layers except the output layer in which the activation function was SoftMax function. We also made use of cross entropy as loss function. To improve the accuracy of the models, we used dropout and regularization techniques in addition to Adam optimizer for carrying out backward propagation of errors. After completing the implementation of the deep learning model, we attempted to find the best dropout and learning rates. To validate the effectiveness of the aspect matching method, we compared the results of SVM [11], Random Forest [2], and FNN. The number of classes in the data set is 6. Through cross-validation in the training step, all sentences were divided into five run sets. Each model had been first trained with the four run sets and then classified each sentence in the rest set to one of the six aspects. Changing the order of the run sets, we performed the train and test steps five times, and measured the average accuracy, precision, recall, and F_1–score of the models. Each model was in standalone executed in a high-performance workstation server with Intel Xeon 3.6GHz CPU with eight cores, 24GB RAM, 2TB HDD, and TITAN-X GPU with 3,072 CUDA cores, 12GB RAM, and 7Gbps memory clock.

For the evaluation metric, we used accuracy, precision, recall, F_1–score measures that have been widely used in IR community. To measure the precision and recall values of a classification model, we first consider a confusion matrix of classes $M_{i,j}$, where each row of the confusion matrix represents predicted class, while each column represents actual class. n is the number of classes. True positive, False positive, and False negative in each class are represented as Eq. (3).

$$\begin{aligned} &\text{True positivie}_i = M_{i,i} \\ &\text{False positivie}_i = \sum_{k=1}^{n} M_{i,k} | k \neq i \\ &\text{False negative}_i = \sum_{k=1}^{n} M_{k,i} | k \neq i \end{aligned} \tag{3}$$

Based on Eq. (20), the precision, recall, and F_1–score (Harmonic mean between precision and recall) are defined as:

$$\begin{aligned} &\text{Precision} = \sum_{k=1}^{n} \frac{\text{True positivie}_i}{\text{True positivie}_i + \text{False positivie}_i} \\ &\text{Recall} = \sum_{k=1}^{n} \frac{\text{True positivie}_i}{\text{True positivie}_i + \text{False negative}_i} \\ &F_1\text{–score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\{\text{Precision} + \text{Recall}\}} \end{aligned} \tag{4}$$

Table 2. Experimental set–up for the used models

Methods	Experimental set-up
SVM	Through many experiments, the optimal trade-off value between training error and margin was selected in each data set
Random Forest	Through Many experiments, the optimal # of trees in the forest & max depth of the tree were selected in each data set
FNN	Batch size=50, Adam optimizer(learning rate=0.01), dropout rate=0.5, 5 hidden layers H_1, H_2, H_3, H_4, and H_5 – H_1 contains 1,000 units; and H_2 contains 800 units;5 hidden layers H_1, H_2, H_3, H_4, and H_5 – H_1 contains 1,000 units; and H_2 contains 800 units; H_3 contains 600 units; H_4 contains 400 units; H_5 contains 200 units

5.2. Experimental Results

Table 3. Accuracy of three aspect matching models based on SVM, Random Forest, and FNN

Aspect	Price	Performance	Design	Service	Quality
FNN	95.7	85.2	93.9	94.6	86.4
SVM	97.3	73.4	89.6	96.1	84.1
Random Forest	96.2	70.6	88.6	95.6	84.2

Result of Aspect Matching Method Table 3 summarizes the average accuracy scores of the three aspect matching models based on SVM, Random Forest, and FNN. By and large, the average accuracy values are high for all aspects. For example, the accuracy of the performance aspect is at least 70.6% in Random Forest. In the price aspect, the accuracy of SVM is up to 97.3%. In three aspects such as performance, design, and service, FNN outperforms both SVM and Random Forest. Interestingly, we observed that the deep learning model like FNN is better than the conventional learning models such as SVM and Random Forest in the aspects including many sentences. In contrast, the price and service aspects have the small number of sentences. In these aspects, SVM is better than the deep learning model. However, the gap of the accuracies in the different learning models is not large. In the data set, a relatively large number of sentences are related to the performance and quality aspects. In general, many sentences in such aspects are often ambiguous because they may be semantically interpreted to other aspect. Thus, developing more intelligent aspect matching models is still challenging and there is room to improve the accuracy of the best learning models.

Sentiment Analysis of the Proposed Method Figure 6 shows the average accuracy, precision, recall, and F_1–scores of the proposed method, comparing to the baseline method that is the typical lexicon–based approach in the sentiment analysis. To find the preference for a particular product, the baseline approach collects (1) review posts, which are related to the product, from several product review web sites; (2) extracts sentences in the collection after the pre–processing step such as stemming and removal of stop words is performed; (3) classifies the polarity (either positive or negative sense) of each sentence based on a sentiment lexicon; and (4) estimates the positive and negative ratios of the product by dividing the total numbers of the positive and negative sentences by the total number of the sentences in the collection. Furthermore, the baseline approach automatically finds important sentences including the positive and negative meaning to/against the product.

As a motivated example, given a product like Hyundai Sonata, customers often want to see the summary note including what positive points are and what negative points are in the 'car design' aspect. They also want to gain more useful information regarding other aspects such as 'car quality,' 'car performance,' and 'car service.' Such an information will enable customers to make good choice when they attempt to purchase their brand–new cars. In addition, car makers will be able to figure out the public's preferences and positive/negative points for new models on market. In the near future, the weak points of

the models will be improved by the sentiment analysis. For this, the baseline approach computes the sentiment score of each sentence and then selects top–k sentences with the highest positive and negative scores. In the figures, the experimental results show that the proposed method outperforms the baseline method in all evaluation metrics. For instance, the average accuracy scores of the baseline method are 75.7% and 68.1% in positive and negative sentences, while those of the proposed method are 79.9% and 77.9% in positive and negative sentences. This indicates that the proposed method improves about 5% and 14% accuracies, compared to the baseline method. Similarly, the average F_1–scores of the baseline method are 67.8% and 68.8% in positive and negative sentences, while those of the proposed method are 75.8% and 79.5% in positive and negative sentences. This implies that the proposed method improves about 12% and 16% F_1–scores, compared to the baseline method. The main reason why the proposed method outperforms the baseline method is that three dimensions (word, sentence, and aspect–levels) are considered to find the reputation tendency and level. In addition, the proposed word–level method considers both importance and sentiment orientation of words, while the baseline method focuses only on measuring the sentiment orientation of words. Another reason is because the proposed method aggregates additional information about the importance of sentences in order to determine the reputation tendency and level. Besides, through the aspect matching method, because most sentences are first categorised to the right aspect, the rest methods have a little chance to get confused to estimate the sentiment scores of the sentences.

(a) Accuracy (b) Precision

(c) Recall (d) F_1–score

Fig. 6. Comparison of the proposed method to the existing lexicon–based approach

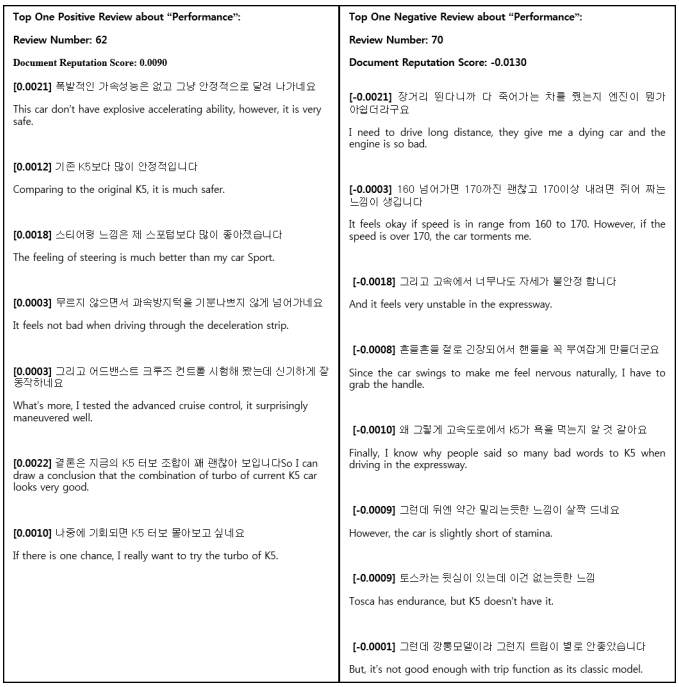


Fig. 7. Top–1 positive and negative review documents

A Case Study of Top–k informative review documents For each aspect, both enterprise executives and customers would like to know the summary of the detailed reviews. If they go over the review summary, they can know the reasons why customers really like the product and what inconvenient points exist to be improved. The proposed method provides top–k documents of the most informative reviews. To validate whether top–k informative reviews are really useful for producers and consumers, we conducted a case study of K5 in Kia motors.

The left figure in Figure 7 shows the top-1 document of positive reviews in the aspect of performance. The identifier of the review document is 62 and the document reputation score is 0.009 that is the sum of the scores of the six sentences in the document. Each sentence also shows the reputation score estimated by our proposed method. For instance, 0.0021 is the reputation score of the first sentence – "This car don't have explosive accelerating ability, however, it is very safe." The top–1 review document contains positive but yet informative meanings. Similarly, the right figure in Figure 7 shows the top–1 document of negative reviews in the same aspect. The top-1 review document contains negative but yet informative meanings. These results clearly show that the top-1 review documents are considerably informative. These review documents will help both producers and consumers figure out the detailed pros and cons of the product that they really want to know in the marketing research.

A User Study and Statistical Verification To validate the effectiveness of our proposed method, we first had interviewed with 30 volunteers who had nothing to do with the authors in this article and are willing to respond to this survey. For each aspect, each interviewee took a look at five sentences chosen at random which are related to the reputation level generated by the proposed method. The interviewee chose one of (i) agree, (ii) disagree, and (iii) N/A to see how much he/she agrees to the results. Figure 8 illustrates the survey results of the six aspects. Y-axis indicates the ratios of agree, disagree, and N/A answers from all interviewees. In the figure, it is obvious that the majority of interviewees agreed to the reputation level, especially in the aspects of design, performance, and service, while it seems that more people disagreed to the reputation level in the quality aspect.

Aspect	PRR	Reputation Level
Price	49.6%	Negative Level 1
Performance	50.4%	Positive Level 1
Design	63.2%	Positive Level 2
Service	20.8%	Negative Level 3
Quality	41.6%	Negative Level 3

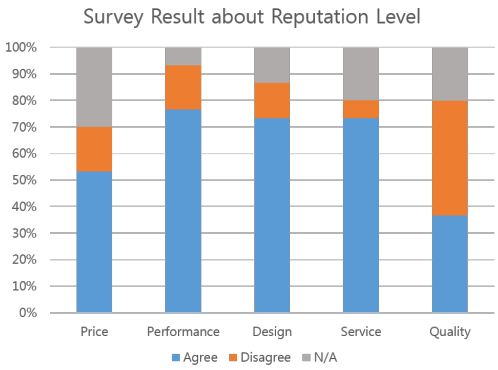


Fig. 8. Results of user study

In addition, we conducted additional survey for top–k informative documents of reviews. In the performance aspect, we prepared top–1 review documents retrieved by the baseline method and the proposed method and showed them to 30 interviewees who gave a score in range from 1 to 5 to each selected document to see how informative it is. We conducted the significance test using IBM-SPSS Statistics 21 and Figure 9 shows the statistical results. We compared the proposed method to the baseline method. When the significant level is 0.05, the null hypothesis H_0 is no statistical difference between the two methods and the alternative hypothesis H_1 is the significant difference between them. According to our Levene's test and t–test results, H_1 is accepted, indicating that the proposed method is statistically different from the baseline method because the p–value is extremely close to 0 and smaller than the significance level. In addition, the interviewees thought that the proposed method is better because the mean score of the proposed method is higher than the baseline method.

6. Concluding Remarks and Future Work

In this work, we propose a novel method of determining the reputation tendency and level and selecting top–k informative review documents about a particular product. This

Group Statistics

		Method	N	Mean	Std.Deviation	Std.Error Mean
Score	1. Baseline	1	30	5.63	1.159	.212
	2. Proposed method	2	30	7.60	1.163	.212

Group Statistics

		Levene's Test for Equality of Variances		Std.Error Mean						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Score	Equal variances assumed	.088	.767	-6.561	58	.000	-1.967	.300	-2.567	-1.367
	Equal variances not assumed			-6.561	57.999	.000	-1.967	.300	-2.567	-1.367

Fig. 9. Statistical test results

product reputation mining approach can help both producers and consumers understand the product well. Unlike the existing lexicon–based approach, our proposed method is based on three dimensional points of word–level, sentence–level, and aspect–level views. In each level, the sentiment orientation of the product is quantified in addition to the consideration of the importance of words and sentences. In addition, the aspect matching process can be helpful in measuring the sentiment orientation of the product. To the best of our knowledge, our method is new, compared to the existing lexicon–based approach. Our experiment results show the the proposed method outperforms the baseline method and we also validated the proposed method through user study and statistical verification tasks.

For our future work, we have a plan to develop a web–based prototype system for the demonstration. We will also apply our method to other domains like smart phones and cosmetic products. Finally, we will propose an automatic method of mining main aspects about a particular product.

Acknowledgments. This work was supported by the National Research of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2019R1F1A1060752) for Byung-Won On, and by the Ministry of Trade, Industry Energy (MOTIE, Korea) under Industrial Technology Innovation Program, No. 10063130 and MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2019-2016-0-00313) supervised by the IITP(Institute for Information communications Technology Promotion) for Gyu Sang Choi.

References

1. Bhonde, R., Bhagwat, B., Ingulkar, S., Pandc, A.: Sentiment analysis algorithms based on dictionary approach. International Journal of Emerging Engineering Research and Technology 3(1), 51–55 (2015)
2. Blondel, M.: Random forest classifier. In: http://scikit-learn.org/.../sklearn.ensemble.RandomForestClassifier.html (2017)
3. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)

4. Canales, L., Strapparava, C., Boldrini, E., Martinez-Barco, P.: A bootstrapping technique to annotate emotional corpora automatically. In: Proceedings of IEEE International Conference on Data Science and Advanced Analytics (DSAA 2016), Montreal, Canada. IEEE (October 17–19, 2016)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
6. Google: Tensorflow. In: https://www.tensorflow.org/ (2018)
7. Grimmer, J., Stewart, M.B., Alvarez, M.: Text as data: The promise and pitfalls of automatic content analysis methods for political texts. Political Analysis 21(3), 267–297 (2013)
8. Hu, Y., Chen, Y., Chou, H.: Opinion mining from online hotel reviews – a text summarization approach. Information Processing & Management 53(2), 436–449 (2017)
9. Hwang, J., Ko, Y.: A korean sentence and document sentiment classification system using sentiment features. Korean Institute of Information Scientists and Engineers 14(3), 336–340 (2008)
10. Jin, W., Hung, H.: A novel lexicalized hmm-based learning framework for web opinion mining. In: Proceedings of the 26th International Conference on Machine Learning (ICML 2009), Montreal, Canada. ICML (June 14–18, 2009)
11. Joachims, T.: Support vector machine. In: https://www.cs.cornell.edu/people/tj/svm_light/ (2014)
12. Khose, N., Dakhode, V.: Product reputation analysis system based on partial supervised word alignment model. International Journal of Science and Research 5(8), 169–173 (2016)
13. Ko, M.: Semantic classification and weight matrices derived from the creation of emotional word dictionary for semantic computing. In: Proceedings of Emotion and Sentiment Analysis Workshop (ESA 2016), Portoroz, Slovenia (May 23, 2016)
14. LeCun, Y., Bottou, L., Orr, G.B., Muller, K.R.: Efficient backprop. In: Proceeding Neural Networks: Tricks of the Trade, this book is an outgrowth of a 1996 NIPS workshop. pp. 9–50. NIPS (1996)
15. Leshno, M., Vladimir Ya, L., Pinkus, A., Schocken, S.: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks 6(6), 861–867 (1993)
16. Medhat, W., Hassan, A., Korashy, H.: Sentiment analysis algorithms and applications: A survey. Ain Shams Engineering Journal 5(4), 1093–1113 (2014)
17. Porter, M.: The porter stemming algorithm. In: https://tartarus.org/martin/PorterStemmer/index.html (2006)
18. Resnick, P., Kuwabara, K., Zeckhauser, R., Friedman, E.: Reputation systems. Communication of the ACM 43(12), 45–48 (2000)
19. Saif, H., He, Y., Fernandez, M., Alani, H.: Contextual semantics for sentiment analysis of twitter. Information Processing & Management 52(1), 5–19 (2016)
20. Skymind: Deeplearning4j. In: https://deeplearning4j.org/word2vec (2017)
21. Steinberger, J., Ebrahim, M., Ehrmann, M., Hurriyetoglu, A., Kabadjov, M., Lenkova, P., Steinberger, R., Tanev, H., Vazquez, S., Zavarella, V.: Creating sentiment dictionaries via triangulation. Decision Support Systems 53(4), 689–694 (2012)
22. Teng, Z., Vo, D., Zhang, Y.: Context-sensitive lexicon features for neural sentiment analysis. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), Austin, Texas (November 1–5, 2016)
23. Wang, J., Yu, L., Lai, K., Zhang, X.: Dimensional sentiment analysis using a regional cnn-lstm model. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), Berlin, Germany (August 7–12, 2016)
24. Wikipedia: Stop_words. In: https://en.wikipedia.org/wiki/Stop_words (2018)
25. Wood, I., Ruder, S.: Emoji as emotion tags for tweets. In: Proceedings of Emotion and Sentiment Analysis Workshop (ESA 2016), Portoroz, Slovenia (May 23, 2016)
26. Yan, J., Turtle, H.: Emocues–28: Extracting words from emotion cues for a fine-grained emotion lexicon. In: Proceedings of Emotion and Sentiment Analysis Workshop (ESA 2016), Portoroz, Slovenia (May 23, 2016)

Zhehua Piao received his Master degree in Department of Computer Science and Engineering, Korea University, Seoul, South Korea. His recent research interests are around Data Mining and Machine Learning, mainly working on Product Reputation Mining, Opinion Mining, Sentiment Analysis and Sentiment Lexicon Construction.

Sang-Min Park is currently attending the Master program in Department of Software Convergence Engineering, Kunsan National University, Gunsan-si, Jeollabuk-do, Korea. His recent research interests are around Machine Learning and Data Mining, mainly working on AI-based Text Mining, Opinion Mining and Korean Sentiment Lexicon Construction.

Byung-Won On received his PhD degree in Department of Computer Science and Engineering, Pennsylvania State University at University Park, PA, USA in 2007. Then, he worked as a full-time researcher in University of British Columbia, Advanced Digital Sciences Center, and Advanced Institutes of Convergence Technology for almost seven years. Since 2014, he has been a faculty member in Department of Software Convergence Engineering, Kunsan National University, Gunsan-si, Jeollabuk-do, Korea. His recent research interests are around Data Mining and Databases, mainly working on AI-based Text Mining and Big Data Management Technologies. He is the corresponding author and can be contacted at: bwon@kunsan.ac.kr

Gyu Sang Choi received his PhD in Computer Science and Engineering from Pennsylvania State University. He was a research staff member at the Samsung Advanced Institute of Technology (SAIT) for Samsung Electronics from 2006 to 2009. Since 2009, he has been with Yeungnam University, where he is currently an associate professor. He is now working on non-volatile memory and storage systems, whereas his earlier research mainly focused on improving the performance of clusters. He is a member of ACM and IEEE. He is the corresponding author and can be contacted at: castchoi@ynu.ac.kr

Myong-Soon Park is Professor of Department of Computer Science and Engineering, Korea University, Seoul, South Korea. He received his BSc in Electronics Engineering from Seoul National University, an MSc in Electrical Engineering from the University of Utah in 1982, and a PhD in Electrical and Computer Engineering from the University of Iowa in 1985. He was an assistant professor at Marquette University from 1985 to 1987.1 and at Postech from 1987.2 to 1988.2. Since 1988.3 he has been an assistant, associate and full professor at Korea University until now. Professor Park was the chair of the SIG on parallel processing of KIISE (1997-2000) and has been on program committees for various international conferences. His research interests include sensor networks, internet computing, parallel and distributed systems, and mobile computing.

Received: July 3, 2018; Accepted: May 20, 2019.

Computer Science and Information Systems 16(2):381–407 https://doi.org/10.2298/CSIS180205001L

Goal-oriented Dependency Analysis for Service Identification ⋆

Jiawei Li[1], Wenge Rong[2], Chuantao Yin[1], and Zhang Xiong[2]

[1] Sino-French Engineer School, Beihang University, Beijing 100191, China
{jiaweili, chuantao.yin}@buaa.edu.cn
[2] School of Computer Science and Engineering, Beihang University, Beijing 100191, China
{w.rong, xiongz}@buaa.edu.cn

Abstract. Highly mature service-oriented architecture systems have great flexibility and reusability, and can align business processes and information technologies with high quality. Service identification plays a key role in this respect. Further, of the different methods employed, the most popular and preferred is process-oriented service identification. However, the absence of dependency analysis in the business process management domain remains a challenge for the quality of future systems. In this paper, we propose a goal-oriented dependency analysis for service identification via business process modeling. In our analysis solution, we apply a dependency tree featuring the relationships among requirements. The dependency relations are analyzed to create business processes via scenarios comprising requirements and process fragments.

Keywords: Service Oriented Computing, Service Identification, Business Process, Dependency

1. Introduction

Aligning business processes and information technology (IT) is an important strategy during a company's development. The results are irreplaceable in the resultant information system architecture [2]. To manage this alignment with different IT implementations, several solutions have been proposed. Service-oriented architectures (SOA) increase versatility and flexibility within a company [54,11,41]. To benefit from SOA, it is essential to define its governance [25]. This becomes a benchmark for justifying whether a given SOA system has achieved its goal. It is nearly impossible to build a perfect SOA system on the first attempt. Therefore, the maturity level and the current state of the system must be analyzed. To this end, several methods have been proposed in the literature. For example, the Combined SOA Maturity Model provides a 7-level maturity process for SOA systems [45]. Similarly, the Independent SOA Maturity Model offers a 5-level process that provides a pathway for SOA systems to become more flexible and mature [42].

A fundamental requirement for SOA governance when pursuing business–IT alignment is fulfilling the need for reusable services in the system [2]. Achieving the proper granularity of a specific service has a significant effect on the reusability of the whole system [17]. As a first and fundamental phase of the management of the SOA's life-cycle,

⋆ Corresponding author: Wenge Rong

service identification helps guarantee the business–IT strategy alignment by communicating business-related issues from an IT perspective [6]. The outcome of this phase influences not only the alignment between strategies [44], but also the development of future systems [35].

Currently there are three main strategies used to identify services within SOA: bottom-up, meet-in-the-middle, and top-down [5]. The top-down strategy is the most popular and most widely used [19]. Of the different kinds of top-down methods, process-driven service identification addresses alignment [16]. Process-oriented solutions for service identification capture functional business requirements. However, non-functional requirements (NFR) are important in business-process modeling, because it provides associated restrictions and constraints [1]. Maintaining the awareness of such dependencies is a challenge, and is helpful for detecting possible conflicts during the early stages [40]. It is difficult to develop a good SOA for complex systems when the complex relations between services are not fully considered [33]. In these studies, it was argued that, when extracting services from business processes, non-functional dependencies should also be assigned importance levels to increase the dependability of identified services.

The degree of dependency between requirements has been proven to have a significant impact on future defects [51]. Moreover, when complexity increases, the number of system errors increases significantly. If we underestimate the importance of dependency, it may result in different bottlenecks and blockages in workflows [47]. Moreover, the idea of services with high adaptability to business changes focuses on managing the dependent relations between business requirements and IT realization [46]. Consequently, it is important to precisely catalogue the dependency-analysis methods.

To solve the dependency-detection problem, many methods have been proposed. User requirement notation (URN) was proposed to provide a more powerful process-modeling language that focuses on dependencies by including goal-dependency management [40]. The authors argued that three perspectives should be guaranteed to achieve this goal: process modeling, goal dependency management, and goal/process traceability. It is thus essential to develop goal-dependency management and goal/process traceability. Whereas URN is powerful with respect to dependencies, it is difficult to implement because of the lack of a suitable design pattern. In the literature, business process management notation (BPMN) is a more user-friendly and popular tool, owing to its graphic presentation [55,43].

Other solutions have been employed to solve this problem by focusing on requirement dependencies [51]. One dependency-detection solution is goal-oriented requirement engineering, which usually applies a model-oriented thinking process [37]. Another approach is i*, which is a pure dependency analysis language proposed for all kinds of possible dependencies [55,15]. In the latest i* model, iStar 2.0, [12], the language was standardized. As a model language, iStar 2.0 proposed relation types without quantitative values to evaluation relations.

Several other model-oriented requirements-engineering methods have been proposed. NFRs are typically more representative of user behavior [36]. However, the logical relationship to business goals is not included. Therefore, Knowledge Acquisition in Automated Specification of Software Systems (KAOS) was proposed to solve this problem [26]. Both NFR [36] and KAOS [27] tended to increase the quantification and traceability of the requirements domain during engineering. Alternatively, GoalBPM was an informal

framework for goal/process traceability [24]. Unfortunately, this solution was dependent on an ambiguous definition of effects.

Cooperating with a model-oriented requirement traceability, Cooperative Requirements Engineering with Scenarios (CREWS) can easily obtain scenarios pertaining to requirements [48]. During the development phase of services, business goals and objectives become performance indicators [39]. Scenarios can be used to trace service performances and goal/process traceability.

Dependency analysis has been successful in requirements management, business process management, and service identification [28]. In this study, we integrate dependency with service identification. First, we model the requirements in the form of scenarios in the requirement-acquisition phase, because the business process is another representation of requirements [7]. Then, the scenario is translated into process fragments [13], which become part of the business process. Each fragment represents a candidate service. Finally, services are grouped per the dependency analysis results. By analyzing the dependency among process fragments, this method identifies services with respect to the successful traceability of business goals, and it processes the dependency relation obtained from the requirement analysis.

Extant dependency analysis methods focus on the graphical representation of dependencies between requirements. One example of dependency analysis is the use of key performance indicators to trace requirements [52]. Another example is iStar 2.0 [12], which uses a dependency net for organized dependency. iStar 2.0 proposes different types of dependencies without quantitative evaluation to identify services. To produce a measurable definition of dependency, we propose a goal-oriented dependency analysis for services identification.

The rest of the paper is organized as follows. In Section 2, we introduce the background to service identification and related methods from a process-oriented perspective. In Section 3, we present details of the proposed method. In Section 4, we evaluate and discuss our method using a case study. Finally, in Section 5, we conclude the paper and present possible future work.

2. Related Work

The alignment of a business–IT strategy is important to an organization's success, considering the fierce market competition and different solutions presented in the literature [18]. As an early attempt to use enterprise architecture, ATIS [3] leveraged the Zachmann framework to measure technology alignment [10]. Recently, with the development of SOA, it was lauded as a feasible method of improving IT governance in the business domain [9].

To implement efficient SOA-based applications, one preliminary task is to obtain proper services [4]. A straightforward idea is to use business entities for service identification by analyzing the relationships among entities [35]. Every element of a business is considered a business entity, and those with strong relations are grouped as services. An example of a business entity is the business process, widely adopted in service identification as a top-down oriented solution, owing to the similarity between business and IT processes [5,19].

Generally, a top-down strategy can have two types of inputs: use cases and business processes [20,22]. Compared to business processes, use cases do not consider tasks that have the same function as units [5]. Alternatively, business process-oriented service identification should design proper metrics for coupling and cohesion [49]. This constitutes the bases for different approaches.

One example of a business process-oriented top-down method was proposed by Kim et al., who created services by analyzing and grouping different business processes or workflows with minimum communication between them [21]. The underlying argument was that a service should represent a group of tasks. Thus, there should be less communication to the outside and more centralization. Similarly, Ma et al. classified business processes by weighting different SOA characteristics, such that customers obtained a group of services with balanced characteristics, according to their needs [31]. Because SOA enhances the flexibility and reusability of services per its design principal [23], to balance the contradictory characteristics, the authors proposed matrix achieved the requirements of an information system.

Another process-driven method, P2S, analyzes the data being sent between tasks [4]. This is suitable for solving complex processes, where interoperation is realized by grouping collaborative tasks. By applying a new definition of business value to determine service definitions, P2S provides a solution to combine data analysis and design metrics. By this definition, business value is a product that is created or treated in one department of an organization and then transferred to another. At this step, P2S obtains several candidate services. Then, it uses pre-defined design metrics to group services together. P2S innovatively combines business values and design metrics to calculate services and improve effectiveness. Moreover, this method has proven to be efficient in decreasing errors.

However, most process-driven methods focus on decomposing business processes. A lack of analysis of their dependencies and goals leaves us to face another challenge with respect to quality analysis [52]. In fact, the reliability of SOA systems depends on the existence of a secure architecture for relation management [14]. However, such an information management system would be difficult to analyze [29]. Thus, it is important to consider the dependency between business processes during service identification.

Identifying dependencies in business processes is recognized as a fundamental challenge in the literature. One possible solution is to use URN [40]. Compared to other popular methods in Table 1, URN has high quality in terms of managing dependencies, including business-process modeling in the goal-management domain. However, its design pattern is incomplete for complex situations. To make it suitable for applications. Three essential parts are necessary [40]. It needs a graphical business-processing modeling language; it needs a goal-oriented method for managing requirements; and it needs a method to relate requirement engineering results to business processes.

Several methods are employed to manage goal-oriented requirements [50], their traceability, and their dependencies. Koliadis et al. proposed the GoalBPM framework [24], which linked BPMN with KAOS [26]. This framework controls goal satisfaction during business-process development. Another goal-oriented requirement traceability method is NFR [36], which goes further in terms of analyzing non-functional requirements and their relations. By classifying goals at different layers, NFR built a goal-oriented system similar to the KAOS model. Instead of focusing on the logical hierarchy among goals, NFR includes non-functional requirements as soft goals in the dependency tree. Instead of us-

Table 1. Different modeling languages for dependency management.

	BPMN	UML	iStar 2.0	NFR	URN
Sequenceflow	√	√	+/-	+/-	√
Roles	√	√	√	×	√
Activities	√	√	×	×	√
Events	√	√	×	×	√
Process Hierarchies	√	√	×	×	√
Goal Modeling	×	×	√	√	√
Goal Model Evaluation	×	×	×	√	√
Goal/Process Traceability	×	×	×	×	√

ing logic relations, as in KAOS, to analyze dependency relations, the NFR dependency tree focuses on the relationship between soft and functional goals. Another efficient goal-oriented method is iStar 2.0 [55,12]. Based on the analysis of the dependency relations among actors, iStar 2.0 forms self-explained modeling languages for tasks in business processes, which include not only the dependency among goals but also dependencies among actors or tasks. At the goal level, iStar 2.0 proposes refinement relationships for goals. See Table 2. For refinement links, iStar 2.0 defines AND-refinement and OR-refinement relationships. However, it does not propose an evaluation method to measure the degree of dependency.

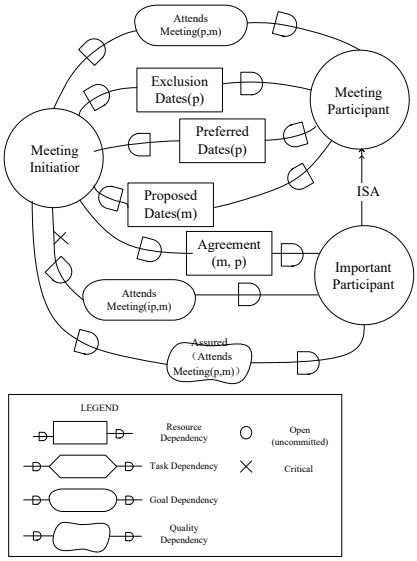


Fig. 1. iStar 2.0 modeling language example

From the literature, business processes have had a close relationship with requirement engineering [7]. Thus, a business process is simply another representation of related and

Table 2. Links between elements in iStar 2.0 model

	Goal	Quality	Task	Resource
Goal	Refinement	Contribution	Refinement	NA
Quality	Qualification	Contribution	Qualification	Qualification
Task	Refinement	Contribution	Refinement	NA
Resource	NA	Contribution	NeededBy	NA

elicited requirements. To model and verify a business process, we must find a suitable requirement engineering method [49,53]. Process fragments [13] are designed to specify an action that is needed to compare business processes with itself in order to manage the overall process. Matching a scenario of requirements to a process fragment helps us understand the logic inside a business process.

3. Dependency-Aware Service Identification

From the above discussion, business process-oriented service identification is promising for SOA-based business–IT alignment, and dependencies among processes should be emphasized simultaneously. There are many dependency detection and analysis tools in the literature, and the methods used to employ their ideas for service identification vary. In this research, we propose a dependency-aware process analysis framework, where we first employ BPMN to model business processes, because BPMN is a powerful extended markup language-oriented machine-friendly language. It enables more choices for gateways and special cases and graphical representations of business processes for ease of understanding [8].

Specifically, we adopted a 3-stage service-identification mechanism for this research. In the requirement-acquisition phase, we recognize requirements as scenarios using the popular CREWS–Scenarios for Acquiring and Validating Requirements [48]. After developing a library of requirements, we develop a KAOS goal-oriented model [26], as a goal-dependency study [32]. We match scenarios using process fragments and. We group service candidates according to the dependency tree, where requirements with dependent relations located in the same root goal have high affinity.

3.1. Requirement Acquisition

The first task for service identification is to define the dependency between different business processes. To analyze the dependency, it is necessary to understand the requirements, because dependencies give rise to conflicts between requirements. To this end, we define requirements as follows:

$$R = \{Id, D, S, Sc\}. \tag{1}$$

In this dependency-analysis method, a requirement, R, is defined by a unique identifier, Id, which serves to guide the relation between requirements and goals when the requirement description changes. For ease of understanding, the description information, D, stored in a unique requirement, should be of a semantic form. It is also possible to trace back to the source of a requirement. The source, S, helps the requirement engineer

review the need for the requirement. The set of scenarios, Sc, included in the definition of a requirement is a representation of traceability management and dependency analysis. CREWS [48] is a model-oriented method employed for scenario construction. In this definition, a unique requirement can have more than one scenario.

After requirements are defined, the next challenge is representing the dependency among requirements. In this research, we employ the dependency tree per the goal-oriented requirement engineering principal by combining the logic relation defined in KAOS [26] and the goal's level distributions of NFR [36]. A branch in the requirement dependency tree is defined as follows:

$$K = \{R, Go, Tt, Sr\}. \tag{2}$$

From the definition of a branch, this equation contains information about the requirements parent and child goals. A branch always points from the leaves to the root. There are two kinds of branches: "AND" branches and "OR" branches. Both branches signify the logical relation between sub-goals and goals. The logical relationship between goals helps identify dependencies between sub goals. The rules are defined in the dependency analysis section. The satisfaction coefficient of a dependency branch is given by the dependency relation between the goal and its sub-goal. To obtain the satisfaction level of a goal, we work from the bottom of the dependency tree. The satisfaction level is classified as "satisfied," "weak," or "unsatisfied." A requirement with all of its scenarios satisfied by the business process will have the state, "satisfied." If only some scenarios are satisfied, the relation is "weak." Otherwise, the requirement is "unsatisfied." This satisfaction relation occurs between the parent goal and a sub-goal, and it can be translated as another form of dependency for the destination goal.

According to the definition of "scenario" in [48], we define a scenario as a sequence of events having one possible pathway through a use case containing some actions.

$$Sce_j = \{ev_0, ..., ev_p\}, \tag{3}$$

where two types of scenarios are further defined. The execution scenario is designed for execution. This kind of scenario has a positive effect on the parent goal. A forbidden scenario is a constraint that should not be executed. Forbidden scenarios have negative effects on the parent goal. When we wish to control for the greatest satisfaction of one goal, it is necessary to combine both positive and negative influences of the sub-goals.

In this research, a scenario is formed by events. In [48], an event could specify the system status before or after an actions resulting in a change. To simplify the comparison between scenarios and business processes, we use only the information of the changing state (i.e., event) but not the details of the action needed in the requirements. Therefore, one event can be defined as a set of data with its new state and the information of the changing source. Each dataset has a data object, a state of data, and a changing source.

$$ev = \{Dt_1, ..., Dt_q\} \tag{4}$$

$$Dt_l = \{Do, st, sc\} \tag{5}$$

Another kind of event is the condition for execution. This event only exists for a condition flux or a condition gateway. A condition event contains one condition description line and a chosen condition. With the chosen condition, we can orient the condition with a

certain condition flux. This kind of event helps us discover complex structures of process fragments.

$$ev = \{Cd_1, ..., Cd_q\} \tag{6}$$

The last kind of event is of temporal significance: state of system. If we need to locate a scenario involving the specific state of a system, it can be found in an event. The state of a system is defined as the need of a company. This kind of event can help to not only define the significant time points for the system, but also the waiting-time for the system. The BPMN modeling business process has several special time events requiring time significance (e.g., interrupted events). Interrupted events make the system wait for a period before executing the predefined action.

$$ev = \{St_1, ..., St_q\} \tag{7}$$

In the analysis of the similarity between scenarios and business processes, process fragments are a part of business processes and can be located as follows.

$$PF = \{Id, T, E, F, A, G, L, \Delta\}. \tag{8}$$

A process fragment is connected to a unique requirement. Therefore, it contains the requirement identification. Inside a process fragment, information exists to rebuild a business process section , including the set of tasks, the set of different kinds of associations, the set of gateways, the set of lanes, and the set of data. Depending on the type of BPMN element, each has its own definition, and they differ according to their identification. Therefore, we follow the identification of each element. There are two types of connecting elements: flow and association.

A sequence flow is the basic connecting element in the BPMN language, and it contains information about the source and the target references. A message flow is a special flow that includes additional information about a message sent in the same direction as the flow itself. As with the definition of the flow, it uses data association. The difference between a basic flow and a simple association is whether or not the two connected elements belong to the same participant. If an association is simple, it connects an internal task with one outside the current participant. To trace the data information of a task, we collect information about the data association. A data association has two additional important variables compared to a basic association: $ioSpecification$ and $DataSet$. If data association is linked to the input data, the $ioSpecification$ is "input," and the $DataSet$ is an inputSet. If the data association is linked to the output data, the $ioSpecification$ is "output," and the $DataSet$ is an outputSet.

For ease of management in data information, the process fragment uses Δ as a set of data. Input Data is a data object linked to a data association with $ioSpecification$="input." Output Data is a data object linked to a data association with $ioSpecification$="output." An event shows changes in the state of data or information before and after a task. We can now compare the difference to understand a business fragment. The matching process is described in the next section.

3.2. Scenario Matching

The objective of this step is to locate a process fragment linking the scenario of a business process requirement. A business process, BP, has a similar definition as process fragment, pf:

$$BP = \{T, E, F, A, G, L, \Delta\}. \tag{9}$$

A business process should have at least one start event and one end event. Normally, a business process belongs to a process fragment. However, a process fragment is not always a business process. To manage the dependency of each business process, we define a relation-matching matrix, which maps the business process to the satisfied process fragment in a requirement. The satisfaction process-fragment management matrix linked to the giving business process saves information pertaining to connected requirements. This matrix is defined as:

$$M_i := [Id, pf_1, ..., pf_m]. \tag{10}$$

This matrix is a $1 \times (m + 1)$ matrix and belongs to a specific business process, where the result corresponds to the scenarios of one requirement. If the business process satisfies the scenario of this requirement, the corresponding pf_j equals 1. Otherwise, it is 0. The size of this matrix depends on the number of scenarios processed by the corresponding requirement. The entire management matrix forms the set, $M[n]$. After searching for the corresponding requirements and scenario sets for each M_i, we have a set of requirements linked to the business process, $R[n]$. For each chosen R_i, we have a set of process fragments, $PF_i[m]$, linked to them. For each R_i, we check each scenario, Sce_j. If the sequence of the process is found to match the sequence of events in the scenario, $pf_{ij+1} \in M_i$ is set to 1. Otherwise, it is set to 0.

Comparing a scenario and business process begins with the first event in Sce_j. According to the definition of PF, we can define a $\tau\{F, A, L, \Delta\}$. The elements belonging to F are sequence flows, condition flows, or default flows. Condition flows and default flows are considered special events. For these, we recognize the condition as information inside the data. In other words, when we meet a conditioned gateway, we should match the condition with the existing data content in an event. Otherwise, a task can only have one in and one out. Thus, neither the flow pointing to the task nor the flow leaving the task influences the comparison of scenarios and business processes. If a scenario has found a matching sequence of tasks, the flows in the business process will be succeeded by the process fragment. From the definitions of the relation between output data and a task and its input, we know that all data are linked to a certain task via data association or another association. Consequently, most comparisons consider the difference between associations and lanes.

Events normally occur either before or after a task. In this research, we assume that our process fragment involves a task before an event. However, tasks after the last event are not considered. First, we determine whether the belonging lane of a task is the same as the changing source of the event. A task's belonging lane should be the same as the changing source of the data post event. When an event has more than one changing source, this is possible only when the event occurs after a gateway. When an event has only one changing source for all data, a task can have only one input and one output. Data actions

generally have four states: create, read, update, and delete [4]. We group all actions (e.g., rewrite, fill up, send, and copy) in the update state, which represents operations performed on the data. Therefore, given a task, τ, and several flows, F, associations, A, swim lanes or collapsed pools, L, and some portions of data, D, linked to the tasks, we can determine the matching method for a satisfied scenario, as shown below.

To match an event with on in a business process, if a start event has a message mission, the same message should have at least $data \in ev$ and $data.state =' R'$, with the message being a part of the data. If an end event has a message mission, the same message should have at least data $data \in ev$ and $data.state =' U'$, with the message being a part of the data.

It is more difficult to match an event with a task than to match an event with another event in a business process. The matching rules are proposed depending on the state of the data. For a data event, if the data state is "C," (i.e., data is created during this task), we have

$$data_l.dataObject \in \tau.output. \tag{11}$$

When this piece of data is a message connected to a message association, then this message association is directed outwards. Most importantly, an object that is created during a task should not be found at any time before this task. For a data event with a data state, "U,"

$$data_l.dataObject \in \tau.input \ \& \ data_l.dataObject \in \tau.output. \tag{12}$$

When this piece of input data is a message connected to a message association, then this message association is directed inwards. When this piece of output data is a message connected to a message association, then this message association is directed outwards. Because updating is a complex operation on a piece of data, the detailed definition of the same update action should be defined by the company itself. For a data event with a data state given as "R," we get

$$data_l.dataObject \in \tau.input. \tag{13}$$

For a data event with a data state given as "D," the data situation should be given as an "R" state. However, in this case, we should be sure that this object will no longer be used.

When an event is found to match the data states of two tasks, the task in front of the testing event will be examined if it is in the assumed lane. If so, the task will be a part of the process fragment. Because we consider the business process for a scenario, we will have the result of satisfaction. For a scenario where we find a process fragment that fulfills all scenario events, this scenario is satisfied. Otherwise, it is unsatisfied. To build a process fragment, we ignore the tasks or gateways between two matched tasks and use a simple flow for connection. If the matched tasks have a parallel, inclusive, or exclusive relation between them, the gateway relation should be inherited by the process fragment. After building the matching-process fragment, we obtain several matching matrices for the relation between the business process and requirements.

3.3. Service Grouping

In this phase, we already have a business process linked to requirements with a matching matrix. Because we used the scenario comparison, the location of the requirement should

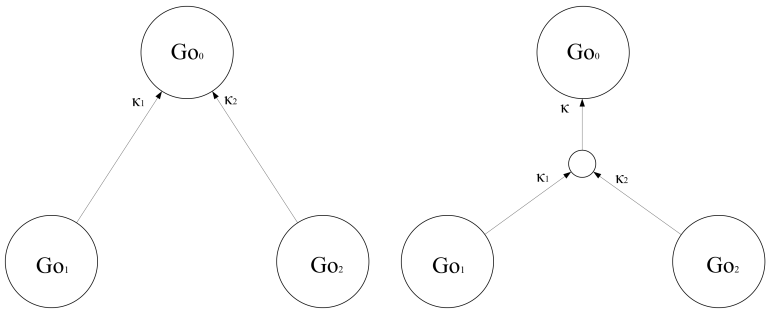
group several tasks together, or they may be located inside one task. A matching scenario forms a candidate service. For candidate services that satisfy the same requirement, we propose that they be grouped together. If a task is identified as being used by several requirements, we recommend grouping services per the minimum connection rule with respect to how loosely coupled they are.

If two process fragments are situated next to one another, we should go through the requirement dependency-relation tree to minimize the dependent relation between two services. The dependent requirement will only be analyzed for one generation, which indicates the leaf generation for the requirement. The resulting service dependency relation is defined as follows:

$$Rs = \{M_0, ..., M_x\}. \tag{14}$$

This is a set of matching matrices for requirements with a satisfaction level of at least "weak". The dependency relation is traced back to the dependency tree by the matching matrix. On each occasion, when a service changes, we trace back to the related requirements for verification, and, according to the goal-oriented model, we obtain a list of possibly impacted services. In the case where there are changes to a specific requirement, services linked to the requirement can be modified rapidly.

To analyze the dependency, we need a goal-dependency coefficient that has a direct relation with the dependency tree. Apart from the branch that points to a goal null, each branch of the dependency tree has a coefficient for the identification of the contribution of a sub-goal, and each coefficient should be between 0 and 1.



(a) Or relation (b) And relation

Fig. 2. Goal relation.

For a goal-dependency relation type, "OR," as shown in Fig. 2(a), we define the coefficient of the dependency as a measurable degree to enable us to understand how well the goal can be satisfied by satisfying the sub-goal. This kind of dependency between parent goal and sub-goal is the level of satisfaction that is contributed by the child goal to the satisfaction of the parent goal. The satisfaction dependency is created by the use of the model for requirement management.

$$\kappa_1, \kappa_2 \in [0, 1] \; and \; \kappa_1 + \kappa_2 \leq 1. \tag{15}$$

The dependency relation between the two sub-goals, $G_D(Go_1, Go_2) = 0$. Because we ignore the other types of dependency, two different sub-goals with the "OR" relation will not influence one another. Then, the dependency relation between Go and Go_1 or Go_2 should be the same value as the coefficient of dependency:

$$G_D(Go_0, Go_1) = \kappa_1,\ G_D(Go_0, Go_2) = \kappa_2. \tag{16}$$

For a goal-dependency relation type, "AND," as shown in Fig. 2(b), the coefficient of dependency should be as follows:

$$\kappa \in [0, 1]\ and\ \kappa_1 \times \kappa_2 = \kappa. \tag{17}$$

In the "AND" relation, two sub-goals have a higher dependency on each another than with the "OR" relation. When we analyze their relations, it is easy to tell if one goal of this type of relation causes a conflict with the parent goal. Their combined effect should also be negative to the parent goal. Therefore, for an "AND" relation, two sub-goals should be at least weakly satisfied for a satisfied goal, Go. Moreover, the dependency relation between the two sub-goals, $G_D(Go_1, Go_2) = 1$, meaning the two sub-goals are not independent of each other and that they should cooperate for the parent goal.

To calculate the goal-dependency relation of a given goal, Go_x, with another goal, Go_y, when we already have a known $\overline{G_D(A, B)}$, we have:

$$G_D(Go_x, Go_y) = G_D(Go_x, A) \times \overline{G_D(A, B)} \times G_D(B, Go_y). \tag{18}$$

Given the definition of the dependency equation between goals, we should find the co-parent for these two goals in the lowest position to obtain their dependency coefficient. Using the special coefficient calculation equation, we predefine if $G_D(Go, Go) = 1$. In other words, one requirement dependent entirely depends on itself, because it shares the same resources with itself.

Using requirement-dependency equations, we can thus conclude a dependency calculation equation, as follows, for two identified services:

$$Y = \frac{\sum_{0 \le i \le u}^{0 \le j \le v} G_D(Go_i, Go_j)}{u \times v}. \tag{19}$$

In this equation, the dependency between services is calculated by the sum of each of their requirements. u and v represent the number of requirements belonging to the two services that are compared.

4. Case Study

To evaluate the service identification method, we performed a case study of the booking process to validate its capability. The booking process contains a basic hotel booking and an entertainment service that is an alternative for customers. Each reservation should be paid for a confirmation of booking. The reservation process is shown in Fig. 3.

To deal with the reservation requirement, the employee of the sales department will show the customer a detailed table of prices. If the customer is not satisfied with the prices and decides against reserving a room or a ticket, the process will end. If they continue to

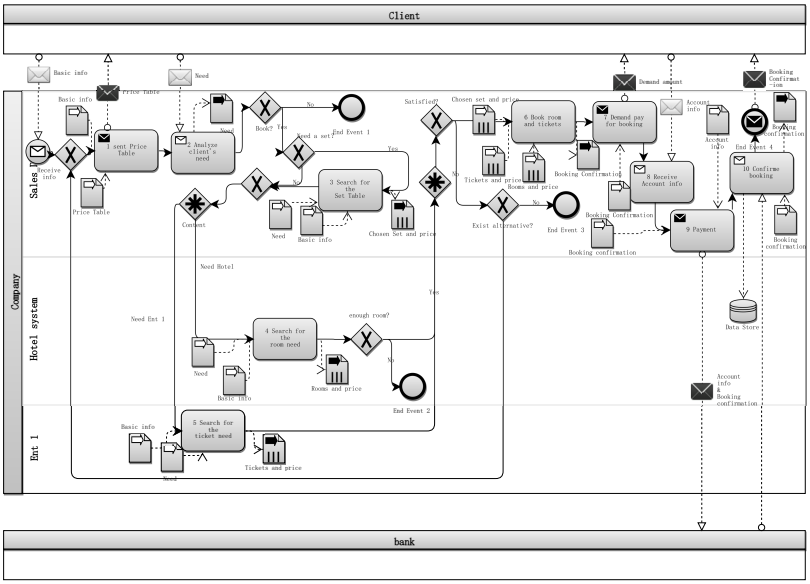


Fig. 3. Booking Process.

the next step, the customer can select from booking only for rooms, only for tickets, or for both. After the booking process, the customer will be either satisfied or unsatisfied with the search result. If they need to look up an alternative, it will be easy to restart from the beginning. When the booking process is completed, the customers are required to pay a reservation fare. Afterwards, a booking confirmation will be sent to the customers.

To calculate the dependency between requirements so that we can reuse the results for obtaining services, we developed a Java-based tool. The first tab of the application is designed for the information of the business process shown in Fig.4.

4.1. Requirement Acquisition

This booking process is linked to several requirements. We have a list of main requirements. The goal-oriented model is built upon the KAOS model proposed in [26]. This goal-oriented model is built on a tree model with a goal-level definition from an NFR requirement management tree and a logic relation definition from a KAOS dependency tree. First, the requirements for this booking process can be derived as follows:

1) R1: Customers want to book hotels or entertainment tickets. 2) R2: Customers want to view the price table. 3) R3: Customers want to receive booking confirmation feedback at the end of the booking. 4) R4: The marketing department wants to promote a different package of tickets to customers. 5) R5: The hotel wants to avoid over-booking. 6) R6: The financial department wants to collect a reservation fee before the booking process ends. 7) R7: Customers want to return to review the price table if not satisfied. 8) R8: The financial department wants to charge booking fee to confirm the booking.

In the tool developed for dependency calculation, we can use requirement management windows to insert a new requirement into the tool, as shown in Fig. 5. In this win-

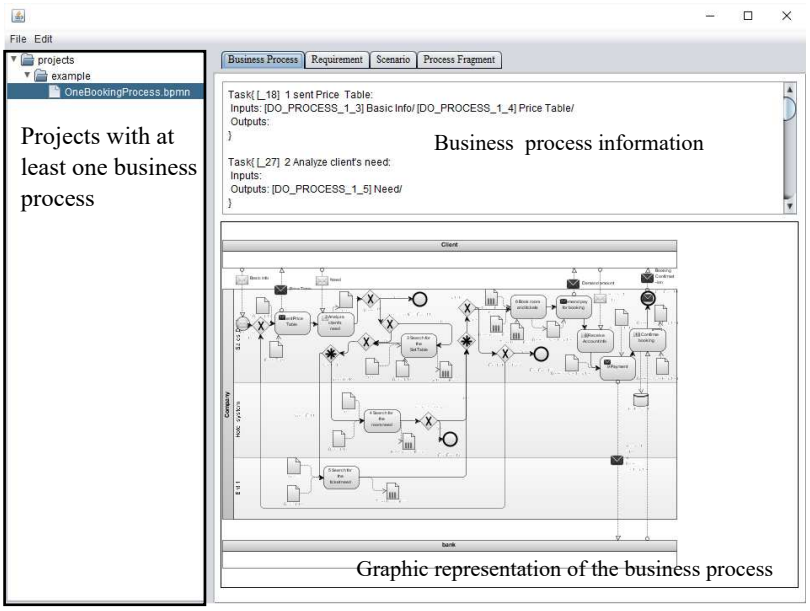


Fig. 4. Dependency Aware Requirement Analysis Tool.

dow, if we create a requirement without pointing it to a parent goal that is not null, we create the goal in terms of strategy levels. If there are choices with respect to the parent goal, a new goal can be chosen from among them. When a goal is connected to its parent goal with a logic AND, it can choose from a list of possible sub-goals of this parent goal with logic AND. Because all sub-goals with logic AND are not connected directly to their parent goal, these choices will influence the dependency analysis process.

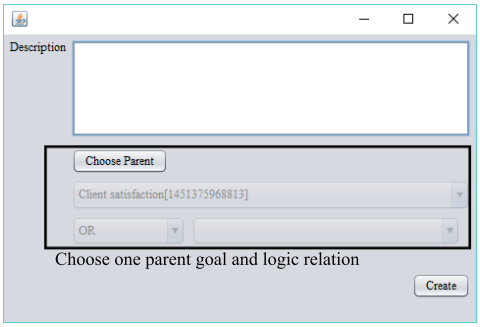


Fig. 5. Requirement-management window used to develop new requirements.

After creating requirements, the dependency tree is automatically built. The dependency in this study is equally distributed. In other words, we consider that all requirements

can fully satisfy their parent goals if satisfied. Then, each sub-goal is equally important according to its logic relation. We can therefore obtain a requirement table using all the information inside, as shown in Fig. 6. In the case of modifying the information of one requirement, we can simply select a row of this table and change the information in the form below it. The dependency tree of this case study is shown in Fig. 7.

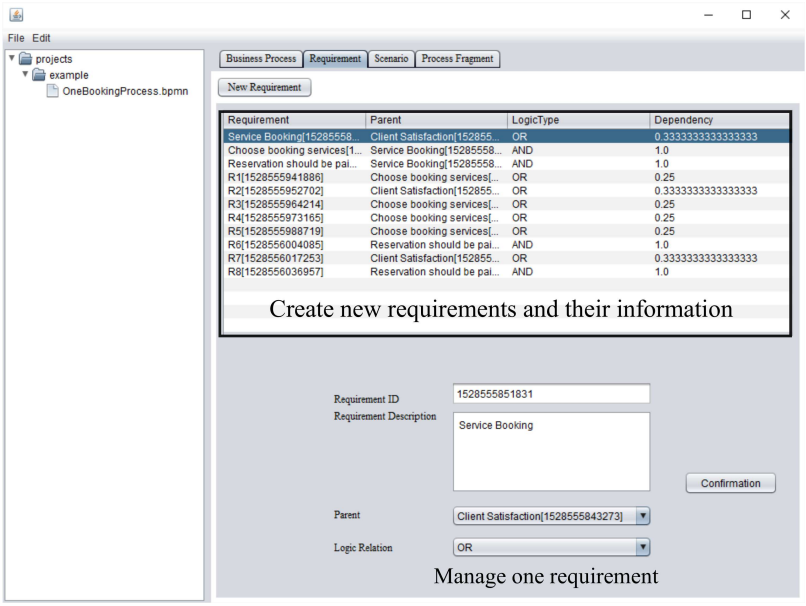


Fig. 6. Requirement-management window for information and editing information.

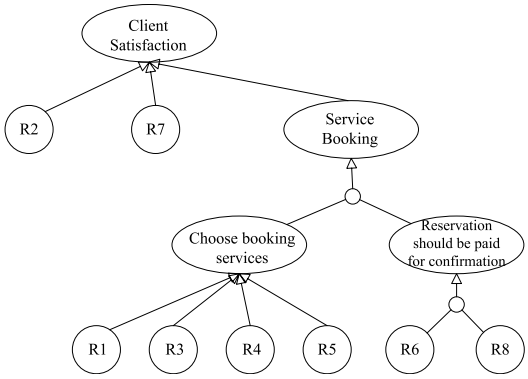


Fig. 7. Goal-oriented model.

Taking requirement R1 as an example, we can derive a requirement and scenarios by using the method proposed in [48], as shown in Fig. 8. We first study requirement R1 and obtain a use case with two possible actions taken by clients. Before these two actions, we can create a “need document. After reading the needs of clients and choosing rooms or tickets for the client, the price shows up, and the process produces a “booking confirmation document.

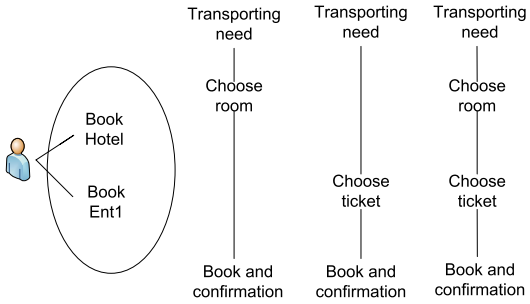


Fig. 8. Scenarios and use case for requirement R1.

Scenario	Events
Scenario 1	Need(C) - Need(R) & RoomChosen(C) - RoomChosen(R) & Confirmation(C)
Scenario 2	Need(C) - Need(R) & Ent1Chosen(C) - Ent1Chosen(R) & Confirmation(C)
Scenario 3	Need(C) - Need(R) & RoomChosen(C) - Need(R) & Ent1Chosen(C) - RoomChosen(R) & Ent1Chosen(R) & Confirmation(C)

Table 3. Scenarios of requirement R1.

According to Table 3, we can use the tab scenario to model the scenario of requirement R1 and the other requirements using data events. Otherwise, we can have the list of other requirements modeled using scenarios shown in Table 4. When we apply all the information related to the scenarios, we obtained in the tool the table shown in Fig. 9

4.2. Scenario Matching

After obtaining scenarios, we can match the process fragment. In the tab, “process fragment, if we apply the “refresh button, we can obtain a simplified version of the process fragment according to certain scenarios. With the help of automatic calculation, we can remodel each process fragment. To continue the analysis, the detailed results of process fragments are shown in Fig. 1 to 8 in the process fragments section in the appendix. Therefore, we can have a group of tasks as a candidate service $\{\tau_2, \tau_4, \tau_5, \tau_6\}$ for requirement R1. Similarly, we have $\{StartEvent, \tau_1, EndEvent_1\}$ for requirement R2, $\{EndEvent_6\}$ for requirement R3, $\{\tau_2, \tau_3, \tau_6\}$ for requirement R4, $\{\tau_4, EndEvent_2\}$ for requirement R5, $\{\tau_7, \tau_8, \tau_9, \tau_{10}, EndEvent_4\}$ for requirement R6,

Requirements	Events
R2	BasicInfo(R) - BasicInfo(R) & PriceTable(U)
R3	Confirmation(U)
R4	Need(R) - BasicInfo(R) & Need(R) & SetChosen(C) - SetChosen(R) & Confirmation(C)
R5	Need(R) & RoomChosen(C) - ”Room: No?”
R6	Confirmation(R) - AcountInfo(R) - AccountInfo(U) - Confirmation(U)
R7	”Exist alternative: No? Yes”
R8	Confirmation(C)

Table 4. Scenarios of requirements R2 to R8.

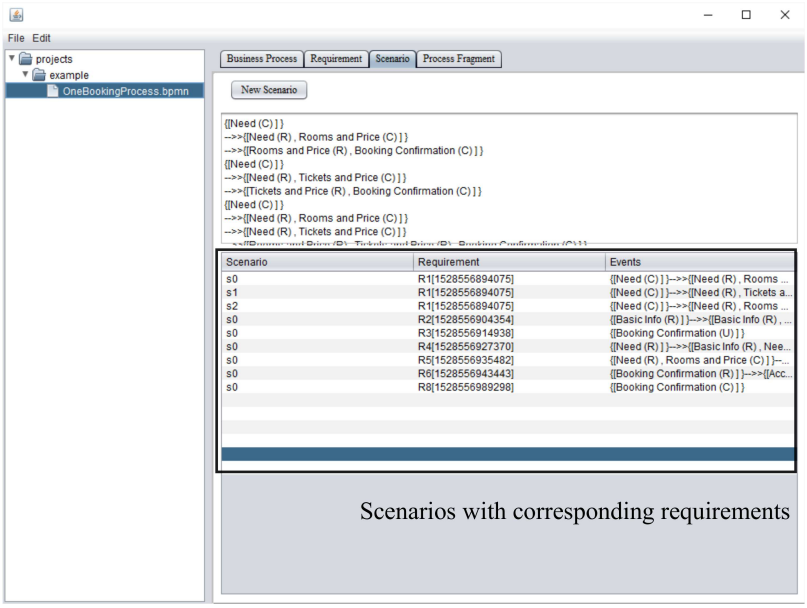


Fig. 9. Information about all scenarios.

$\{StartEvent, EndEvent_3\}$ for requirement R7, and $\{\tau_{10}, EndEvent_4\}$ for requirement R8.

In order to analyze the relation between two services, we calculated the dependency relation between them using the modeling tool shown in Fig. 10. The result is calculated automatically per the definition of dependency. According to the dependency tree, we can classify three main services: {StartEvent, τ_1}, {τ_2, $EndEvent_1$, τ_3, $EndEvent_2$, τ_4, τ_5, $EndEvent_3$, τ_6}, and {τ_7, τ_8, τ_9, τ_{10}, $EndEvent_4$}. With this proposition of candidate services, we can calculate the dependency between any two, and the results are shown in Table 5. According to the table, three services are relatively independent of each other.

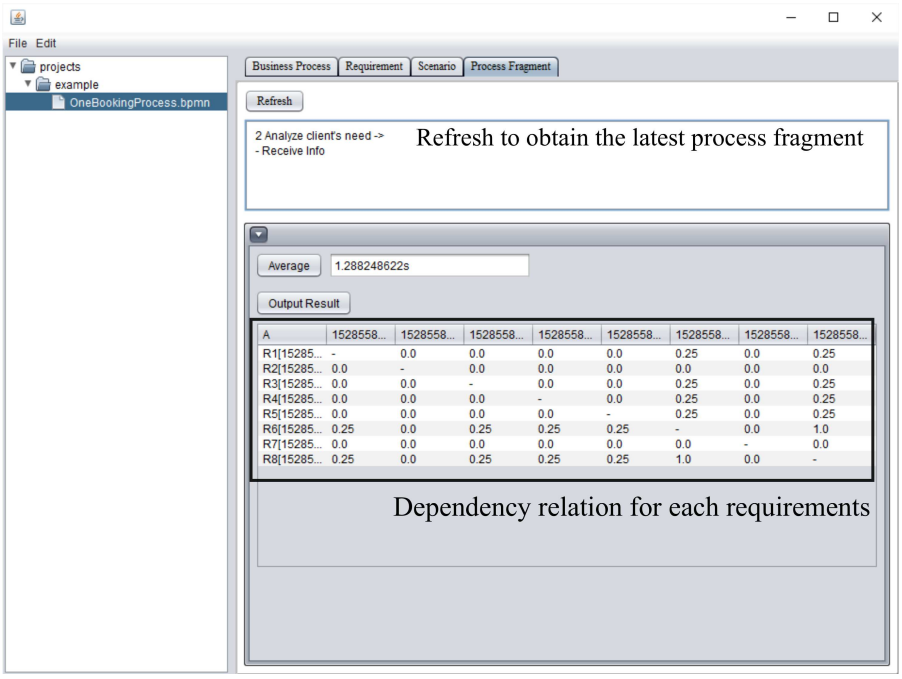


Fig. 10. Dependency relation for services.

Candidate services	Requirements	C1	C2	C3
C1	R_1, R_4, R_5	-	0	0.17
C2	R_2, R_7	0	-	0
C3	R_3, R_6, R_8	0.17	0	-

Table 5. Dependency between candidate services.

4.3. Discussion

To evaluate the proposed method, we compared it to other popular methods, as shown in Table 6. The [35] method was used in an attempt to cluster business entities, but the definition of business entities can change from person to person. Once the business process changes, the standard of business entities will need to be redefined. The [31] method was applied to a workflow. The result can change according to the weight matrix. However, we should not amend the weight matrix. This matrix will be used for different business needs and an amendment will cause a big impact. [21] provided another process-driven method that works by analyzing the Petri net. However, it depends on a rule respecting a minimum communication between services. When a new communication is established, it may influence the belonging location of a task. The structure of services may also change. [4] is the only method that can be tested for our case study. The result shows that R6 is separated. We determined that the reason was the absence of a connection between tasks 7 and 8 and tasks 9 and 10. Therefore, the business process should have enough details to enable the calculation.

The goal-oriented dependency analysis in the service-identification method has not only an advantage in the service-identification phase, but also with respect to the continuity of the life cycle of services. If the business process changes for a short period, all the other methods must redo the calculation. Because the services identified from our method are related to specific requirements, it is not expected that there would a significant change to the structure of services. However, the other method does not guarantee this.

Table 6. Comparison between service-identification methods.

Method	Inputs	Method	Evaluation	Apply to this case?	For future governance in SOA?
Goal-Oriented dependency analysis	Business process in BPMN	top-down	case study and evaluations	Yes 3 services	Yes. No need to redo the calculation. Services are traceable linking to requirements and they change only when requirements change.
[4]	Business process in BPMN	top-down	case study and evaluations	Yes 5 services	No. Should redo the calculation if business process changes.
[21]	Petri Net	top-down	case study and evaluation	No	No. Should redo the calculation if Petri net changes.
[31]	Workflow	top-down	case study	No	No. Should redo the calculation.
[35]	Business Entities	top-down	case study and evaluation	No	No. Should redo the calculation.

From the dependency relation table shown in Fig. 10, we can also get dependency calculation results for any two requirements. Depending on the number of requirements

accumulated in the first step, we will obtain a table of a different size. In this table, we can also verify the time required to obtain the matching process fragment. We then obtain the result shown in Table 7. The times taken to obtain all data events are nearly equal, and the number of events is not expected to change while obtaining results. The searching method used in this tool causes the only dependency of time based on the size of the business process.

Table 7. Testing the matching efficiency.

Requirement/Scenario	First data event	Last data event	Entire scenario	Average
R1sce1	0.023131306 s	0.035481503 s	0.022244722 s	0.020 s
R1sce2	0.038197306 s	0.02403289 s	0.020332213 s	0.022 s
R1sce3	0.020299843 s	0.02483421 s	0.024278812 s	0.022 s
R2	0.020369713 s	0.026578955 s	0.023192885 s	0.025 s
R3	0.020369713 s	0.023752625 s	0.027197117 s	0.022 s
R4	0.020315238 s	0.021352217 s	0.024698419 s	0.021 s
R5	0.020663793 s	0.021249191 s	0.020074053 s	0.021 s
R6	0.020581293 s	0.024200654 s	0.020899847 s	0.021 s
R8	0.020717083 s	0.020313659 s	0.024139469 s	0.020 s

5. Conclusion and Future Work

We proposed a goal-oriented dependency-analysis method for service-identification by finding the business requirements in a business process, realizing the dependency relation of requirements for business processes and services in SOA, and proposing a definition of dependency among services. As shown in the case study, this method can provide another proposed standard for classifying tasks to services before applying design metrics to identify services. We considered the definition of cohesion; the loosely coupling of SOA is closely linked to the requirement. A service that integrates fewer numbers of possible requirements is more specific. A service will be more independent if it is not required cooperate with another service serving the same requirement. Therefore, in this study, we developed a tool to manage requirements and calculate the process fragments. Moreover, with the predefinition of dependency equations, we can easily obtain the dependency among services.

Because there are still a variety of gateways and events in business processes, process fragments face more complex business processes with which they should be matched. Additionally, we plan to study the case where a business process does not fully satisfy a requirement. For example, if the requirement is difficult to fulfill because of limited capability, the important part of the scenario will be satisfied and the rest will be ignored. In this case, it should still be possible to recognize the requirement in a business process.

For future work, service identification will be extended to services with web service definition language so that the identified services can be discovered. Then, by pairing business process with IT processes [20], it will possible to develop a service-identification phase that is more traceable both from requirements and technical perspectives. Furthermore, when applying SOA to the design of Web services, the low frequency with which

services are reused is a challenging task [54]. To decrease the difficulty of finding services, several solutions have been proposed, for which a complete knowledge warehouse appears to be a promising solution [38]. However, it difficult to relocate a particular service from a large service center [30]. Thus, it would be a large amount of management work for the service center. In this paper, we proposed to organize different services based on the requirements management method, because it helps improve the efficiency of managed changes [34]. Therefore, there is a need for further research to develop linkages between requirements and the knowledge-based services center.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China (No. 61472021).

References

1. Aburub, F., Odeh, M., Beeson, I.: Modelling non-functional requirements of business processes. Information & Software Technology 49(11-12), 1162–1171 (2007)
2. Aversano, L., Grasso, C., Tortorella, M.: A literature review of business/it alignment strategies. In: Proceedings of 14th International Conference on Enterprise Information Systems. pp. 471–488 (2012)
3. Avila, O., Goepp, V., Kiefer, F.: ATIS: A method for the complete alignment of technical information systems. International Journal of Computer Integrated Manufacturing 24(11), 993–1009 (2011)
4. Bianchini, D., Cappiello, C., Antonellis, V.D., Pernici, B.: Service identification in interorganizational process design. IEEE Transactions on Services Computing 7(2), 265–278 (2014)
5. Bianchini, D., Pagliarecci, F., Spalazzi, L.: From service identification to service selection: An interleaved perspective. In: Proceedings of Formal Modeling: Actors, Open Systems, Biological Systems - Essays Dedicated to Carolyn Talcott on the Occasion of Her 70th Birthday. pp. 223–240 (2011)
6. Börner, R., Goeken, M.: Identification of business services literature review and lessons learned. In: Proceedings of 15th Americas Conference on Information Systems (2009)
7. Castano, S., Antonellis, V.D., Melchiori, M.: A methodology and tool environment for process analysis and reengineering. Data & Knowledge Engineering 31(3), 253–278 (1999)
8. Chinosi, M., Trombetta, A.: BPMN: an introduction to the standard. Computer Standards & Interfaces 34(1), 124–134 (2012)
9. Choi, J., Nazareth, D.L., Jain, H.K.: The impact of SOA implementation on it-business alignment: A system dynamics approach. ACM Transactions on Management Information Systems 4(1), 3 (2013)
10. Dahman, K., Charoy, F., Godart, C.: Alignment and change propagation between business processes and service-oriented architectures. In: Proceedings of 2013 IEEE International Conference on Services Computing. pp. 168–175 (2013)
11. Dai, W.W., Vyatkin, V., Christensen, J.H., Dubinin, V.N.: Bridging service-oriented architecture and IEC 61499 for flexibility and interoperability. IEEE Transactions on Industrial Informatics 11(3), 771–781 (2015)
12. Dalpiaz, F., Franch, X., Horkoff, J.: istar 2.0 language guide. CoRR abs/1605.07767 (2016)
13. Daniel, F., Casati, F., D'Andrea, V., Mulo, E., Zdun, U., Dustdar, S., Strauch, S., Schumm, D., Leymann, F., Sebahi, S., Marchi, F.D., Hacid, M.: Business compliance governance in service-oriented architectures. In: Proceedings of 23rd IEEE International Conference on Advanced Information Networking and Applications. pp. 113–120 (2009)
14. Delac, G., Silic, M., Srbljic, S.: A reliability improvement method for soa-based applications. IEEE Transactions on Dependable and Secure Computing 12(2), 136–149 (2015)

15. Gonçalves, E., Castro, J., Araújo, J., Heineck, T.: A systematic literature review of istar extensions. Journal of Systems and Software 137, 1–33 (2018)
16. Gu, Q., Lago, P.: Service identification methods: A systematic literature review. In: Proceedings of 3rd European Conference on ServiceWave. pp. 37–50 (2010)
17. Haesen, R., Snoeck, M., Lemahieu, W., Poelmans, S.: On the definition of service granularity and its architectural impact. In: Proceedings of 2008 Advanced Information Systems Engineering, 20th International Conference. pp. 375–389 (2008)
18. Henderson, J.C., Venkatraman, N.: Strategic alignment: Leveraging information technology for transforming organizations. IBM Systems Journal 32(1), 4–16 (1993)
19. Huergo, R.S., Pires, P.F., Delicato, F.C., Costa, B., Cavalcante, E., Batista, T.: A systematic survey of service identification methods. Service Oriented Computing and Applications 8(3), 199–219 (2014)
20. Inaganti, S., Gopala, Behara, K.: Service identification: BPM and SOA handshake. BPTrends (2007)
21. Kim, Y., Doh, K.: Formal identification of right-grained services for service-oriented modeling. In: Proceedings of 10th International Conference on Web Information Systems Engineering. pp. 261–273 (2009)
22. Kim, Y., Doh, K.: Use-case driven service modelling with xml-based tailoring for SOA. International Journal of Web and Grid Services 9(1), 35–53 (2013)
23. Kohlborn, T., Korthaus, A., Chan, T., Rosemann, M.: Identification and analysis of business and software services - A consolidated approach. IEEE Transactions on Services Computing 2(1), 50–64 (2009)
24. Koliadis, G., Ghose, A.: Relating business process models to goal-oriented requirements models in KAOS. In: Proceedings of 2006 Pacific Rim Knowledge Acquisition Workshop on Advances in Knowledge Acquisition and Management. pp. 25–39 (2006)
25. Koumaditis, K., Themistocleous, M.: A detailed framework for SOA governance. International Journal of Systems and Service-Oriented Engineering 5(3), 52–74 (2015)
26. van Lamsweerde, A.: Goal-oriented requirements engineering: A guided tour. In: Proceedings of 5th IEEE International Symposium on Requirements Engineering. pp. 249–262 (2001)
27. Letier, E., van Lamsweerde, A.: Reasoning about partial goal satisfaction for requirements and design engineering. In: Proceedings of the 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering. pp. 53–62 (2004)
28. Li, J., Rong, W., Yin, C., Xiong, Z.: Dependency aware business process analysis for service identification. In: Proceedings of 9th Asia-Pacific Services Computing Conference. pp. 137–152 (2015)
29. Li, Q., Wang, Z., Cao, Z., Du, R., Luo, H.: Process and data fragmentation-oriented enterprise network integration with collaboration modelling and collaboration agents. Enterprise Information Systems 9(5-6), 468–498 (2015)
30. Llinas, G.A.G., Nagi, R.: Network and qos-based selection of complementary services. IEEE Transactions on Services Computing 8(1), 79–91 (2015)
31. Ma, Q., Zhou, N., Zhu, Y., Wang, H.: Evaluating service identification with design metrics on business process decomposition. In: Proceedings of 2009 IEEE International Conference on Services Computing. pp. 160–167 (2009)
32. Maiden, N.A.M., Lockerbie, J., Randall, D., Jones, S., Bush, D.: Using satisfaction arguments to enhance i* modelling of an air traffic management system. In: Proceedings of 15th IEEE International Requirements Engineering Conference. pp. 49–52 (2007)
33. Mayer, S., Wilde, E., Michahelles, F.: A connective fabric for bridging internet of things silos. In: Proceedings of 5th International Conference on the Internet of Things. pp. 148–154 (2015)
34. Mellegård, N., Staron, M.: Improving efficiency of change impact assessment using graphical requirement specifications: An experiment. In: Proceedings of 11th International Conference on Product-Focused Software Process Improvement. pp. 336–350 (2010)

35. Merabet, M., Benslimane, S.M.: A multi-objective hybrid particle swarm optimization-based service identification. In: Proceedings of 1st International Conference on Advanced Aspects of Software Engineering. pp. 52–62 (2014)
36. Mylopoulos, J., Chung, L., Nixon, B.A.: Representing and using nonfunctional requirements: A process-oriented approach. IEEE Transactions on Software Engineering 18(6), 483–497 (1992)
37. Nuseibeh, B., Easterbrook, S.M.: Requirements engineering: a roadmap. In: Proceedings of 22nd International Conference on on Software Engineering. pp. 35–46 (2000)
38. Papazoglou, M.P., van den Heuvel, W., Mascolo, J.E.: A reference architecture and knowledge-based structures for smart manufacturing networks. IEEE Software 32(3), 61–69 (2015)
39. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: State of the art and research challenges. IEEE Computer 40(11), 38–45 (2007)
40. Pourshahid, A., Amyot, D., Peyton, L., Ghanavati, S., Chen, P., Weiss, M., Forster, A.J.: Business process management with the user requirements notation. Electronic Commerce Research 9(4), 269–316 (2009)
41. Raman, A., Bharadwaj, S.S., Mukherjee, J.: Developing soa-enabled service agility capabilities: case studies in services industry. International Journal of Business Information Systems 27(1), 21–44 (2018)
42. Rathfelder, C., Groenda, H.: isoamm: An independent SOA maturity model. In: Proceedings of 8th International Conference Distributed Applications and Interoperable Systems. pp. 1–15 (2008)
43. Salles, G.M.B., Fantinato, M., Barros, V.A., de Albuquerque, J.P.: Evaluation of the strali-bpm approach: strategic alignment with BPM using agreements in different levels. International Journal of Business Information Systems 27(4), 433–465 (2018)
44. Schelp, J., Aier, S.: SOA and EA - sustainable contributions for increasing corporate agility. In: Proceedings of 42nd Hawaii International International Conference on Systems Science. pp. 1–8 (2009)
45. Söderström, E., Meier, F.: Combined SOA maturity model (CSOAMM): towards a guide for SOA adoption. In: Proceedings of the 3th International Conference on Interoperability for Enterprise Software and Applications. pp. 389–400 (2007)
46. Stephan, B., Bauer, T., Reichert, M.: Bridging the gap between business process models and service composition specifications. In: Service Life Cycle Tools and Technologies: Methods, Trends and Advances, pp. 124–153 (2011)
47. Strode, D.E.: A dependency taxonomy for agile software development projects. Information Systems Frontiers 18(1), 23–46 (2016)
48. Sutcliffe, A.G., Maiden, N.A.M., Minocha, S., Manuel, D.: Supporting scenario-based requirements engineering. IEEE Transactions on Software Engineering 24(12), 1072–1088 (1998)
49. Vanderfeesten, I.T.P., Reijers, H.A., van der Aalst, W.M.P.: Evaluating workflow process designs using cohesion and coupling metrics. Computers in Industry 59(5), 420–437 (2008)
50. Vilela, J., Castro, J., Martins, L.E.G., Gorschek, T., Silva, C.T.L.L.: Specifying safety requirements with GORE languages. In: Proceedings of the 31st Brazilian Symposium on Software Engineering. pp. 154–163 (2017)
51. Wang, J., Wang, Q.: Analyzing and predicting software integration bugs using network analysis on requirements dependency network. Requirements Engineering 21(2), 161–184 (2016)
52. Wetzstein, B., Leitner, P., Rosenberg, F., Dustdar, S., Leymann, F.: Identifying influential factors of business process performance using dependency analysis. Enterprise Information Systems 5(1), 79–98 (2011)
53. Xu, L.D., Viriyasitavat, W., Ruchikachorn, P., Martin, A.: Using propositional logic for requirements verification of service workflow. IEEE Transactions on Industrial Informatics 8(3), 639–646 (2012)
54. Yao, J., Tan, W., Nepal, S., Chen, S., Zhang, J., Roure, D.D., Goble, C.A.: Reputationnet: Reputation-based service recommendation for e-science. IEEE Transactions on Services Computing 8(3), 439–452 (2015)

55. Yu, E.S.K.: Towards modeling and reasoning support for early-phase requirements engineering. In: Proceedings of 3rd IEEE International Symposium on Requirements Engineering. pp. 226–235 (1997)

Appendix A. Process fragment

The result of process fragments are shown below from Fig. 1 to 8.

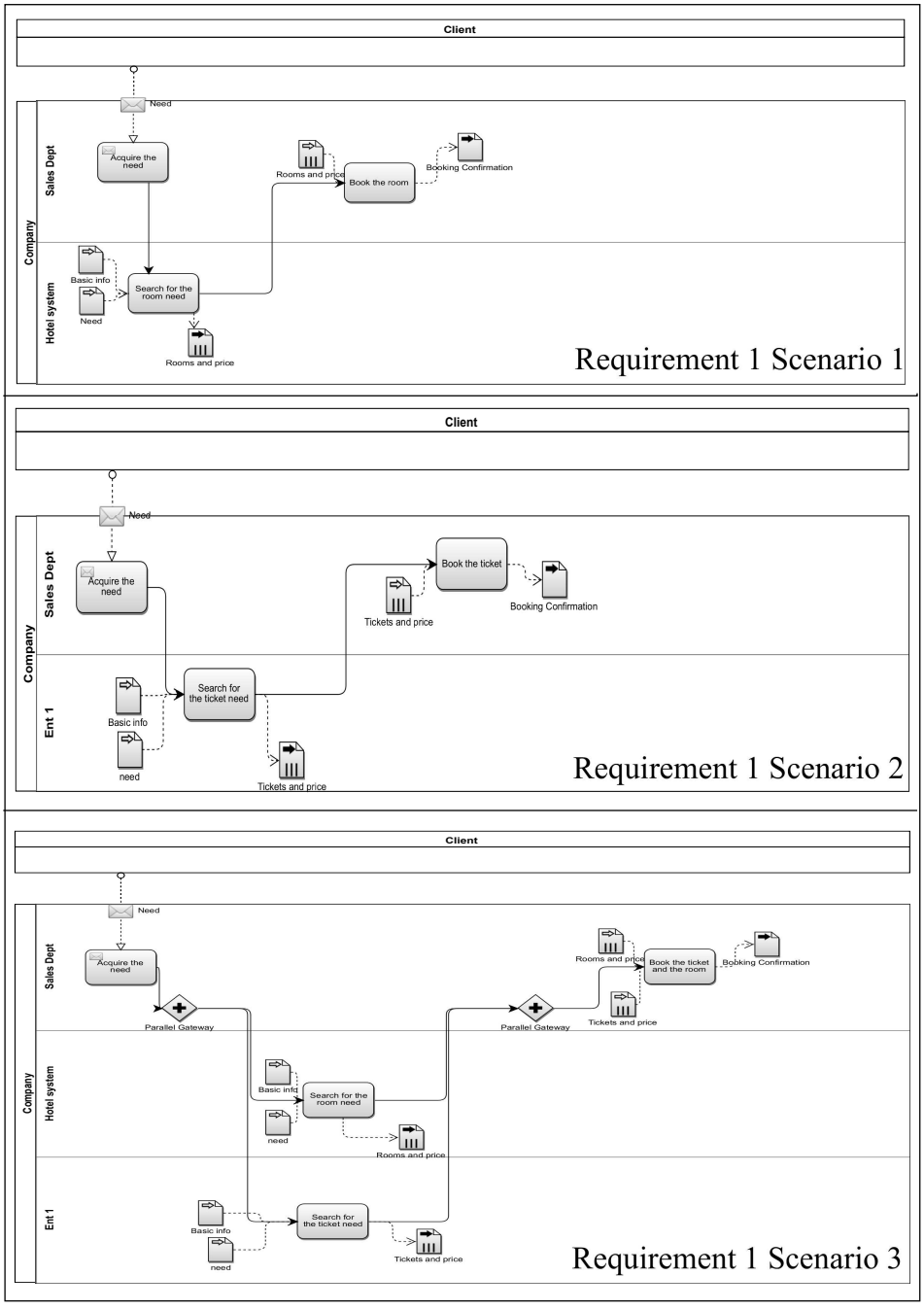


Fig. 1. Requirement R1.

Fig. 2. Requirement R2.

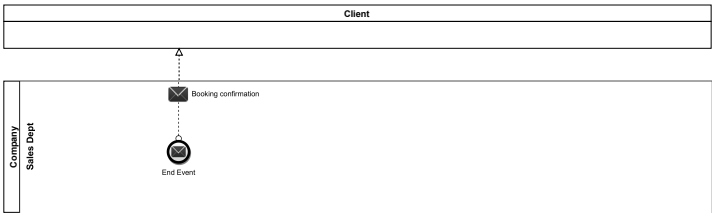


Fig. 3. Requirement R3.

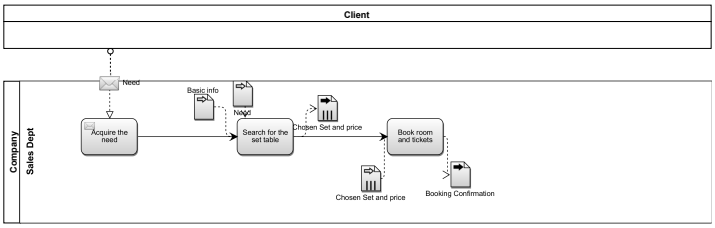


Fig. 4. Requirement R4.

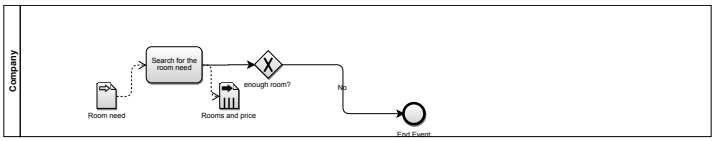


Fig. 5. Requirement R5.

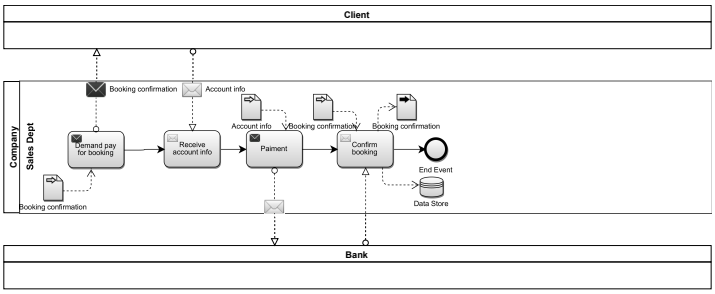


Fig. 6. Requirement R6.

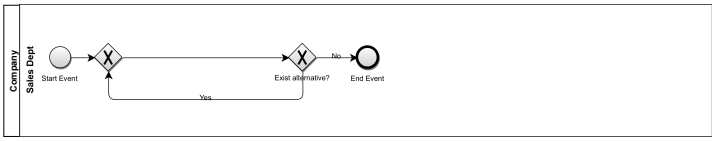


Fig. 7. Requirement R7.

Fig. 8. Requirement R8.

Jiawei Li received her MSc degree from Sino-French Engineering School at Beihang Universit in 2016. Her research interest covers service oriented computing, software engineering and information systems.

Wenge Rong received the B.Sc. degree from the Nanjing University of Science and Technology, China, in 1996, the M.Sc. degree from Queen Mary College, U.K., in 2003, and the Ph.D. degree from the University of Reading, U.K., in 2010. He is currently an Associate Professor with Beihang University, China. He has many years of working experience as a Senior Software Engineer in numerous research projects and commercial software products. His current research interests include machine learning, natural language processing, and information management.

Chuantao Yin received his PhD degree on computer science in 2010 from Ecole Centrale de Lyon. He works as associate professor in Sino-French Engineering School at Beihang University in China. His research activities are focused on human learning, smart learning, smart city, etc.

Zhang Xiong is currently a Professor with the School of Computer Science of Engineering, Beihang University, and the Director of the Advanced Computer Application Research Engineering Center, National Educational Ministry of China. He has published over 200 referred papers in international journals and conference proceedings. His research interests and publications span from smart cities, knowledge management, and information systems. He received the National Science and Technology Progress Award.

Received: February 5, 2018; Accepted: January 15, 2019.

Computer Science and Information Systems 16(2):409–442 https://doi.org/10.2298/CSIS180411002N

Intelligent query processing in P2P networks: semantic issues and routing algorithms

AL Nicolini[1], CM Lorenzetti[1], AG Maguitman[1], and CI Chesñevar[1]

Institute for Computer Science and Engineering (ICIC)
Universidad Nacional del Sur - CONICET, Bahía Blanca, Argentina
{aln, cml, agm, cic}@cs.uns.edu.ar

Abstract. P2P networks have become a commonly used way of disseminating content on the Internet. In this context, constructing efficient and distributed P2P routing algorithms for complex environments that include a huge number of distributed nodes with different computing and network capabilities is a major challenge. In the last years, query routing algorithms have evolved by taking into account different features (provenance, nodes' history, topic similarity, etc.). Such features are usually stored in auxiliary data structures (tables, matrices, etc.), which provide an extra knowledge engineering layer on top of the network, resulting in an added *semantic* value for specifying algorithms for efficient query routing. This article examines the main existing algorithms for query routing in unstructured P2P networks in which *semantic aspects* play a major role. A general comparative analysis is included, associated with a taxonomy of P2P networks based on their degree of decentralization and the different approaches adopted to exploit the available semantic aspects.

Keywords: P2P systems, query routing, network topology.

1. Introduction

A peer-to-peer network (or just P2P network) is a computing model present in almost every device, from smartphones to large-scale servers, as a way to leverage large amounts of computing power, storage, and connectivity around the world. In a P2P network, each peer can act indistinctly as a client and a server and can collaborate in order to share information in a distributed environment without any centralized coordination. These systems are vulnerable to security problems, abuse, and other threats, and consequently, it is necessary to be resilient to different forms of attacks, to have mechanisms to detect and remove poisoned data [20,72], and to distinguish spammers from honest peers [92,122,94]. Despite these issues, P2P networks are widely used for large-scale data sharing, content distribution, and application-level multicast working with a tolerable waiting time for the users [86,97,144].

P2P technologies have demonstrated great potential to support distributed information retrieval. The typical information retrieval problem in P2P networks involves finding a set of documents in the network that are relevant to a given query. To better describe the problem of information retrieval in P2P networks, it is useful to identify a number of salient features, as illustrated in Figure 1. In a P2P information retrieval network each device or node (peer) maintains a collection of documents available to share with other

peers. In order for those peers to interact with each other, several components are required to support search among peers associated with a given query. These components include a routing algorithm, routing tables, indices, and an established protocol that manages the queries that each node can handle [56,48,121]. The routing algorithm determines how a node searches for information by sending query messages to other peers. When a peer receives a query message, it attempts to retrieve relevant documents from its own collection, forwarding as well the query to other peers in the network.

In the context of P2P networks, it is necessary to distinguish between the *physical* network and the *logical* network. The former consists of real physical connections between devices while the latter is a topology that emerges from the peers' interaction. Since the interaction between peers can be guided by their semantic relations, *semantic communities* commonly emerge in logical networks [36,22].

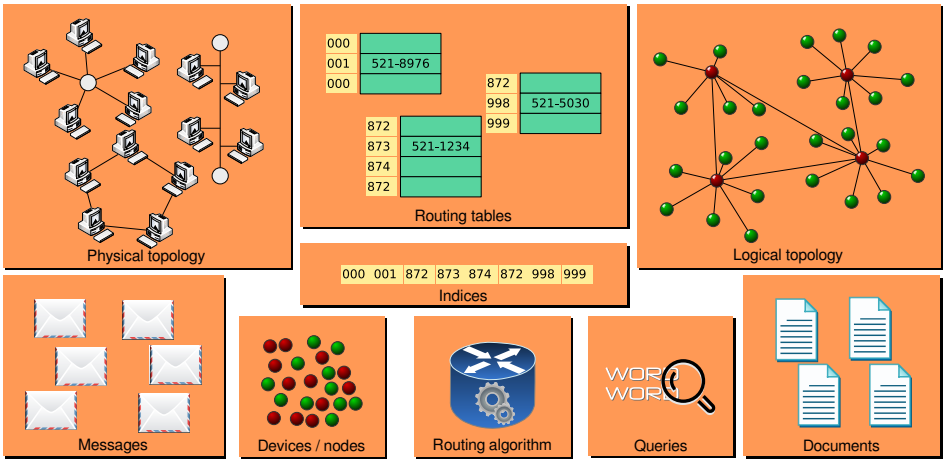


Fig. 1. Conceptual elements which characterize a P2P network.

Different methodologies and techniques for information retrieval on P2P networks have been proposed [129], providing alternative approaches to exploit the concept of social communities on the Internet. Some of the benefits which result from applying P2P information retrieval networks are the following:

- By their very nature, P2P networks do not need a centralized administration, being self-organizing and adaptive (in the sense that peers can enter and leave the network without any external control).
- Peers can have access to several storage and processing resources available from different computers and devices in the network.
- Since pure P2P networks are distributed and decentralized, they tend to be fault-tolerant and with a good load balance for handling network traffic.

Over the years, the Internet has become increasingly restricted to client-server applications. Unfriendly protocols and firewalls are examples of aspects that restrict and limit

the use of Internet. To some extent, P2P technologies can be thought of as a way of returning the Internet to its original cooperative design, in which every participant creates as well as consumes [101]. The emerging P2P networks could thus empower almost any group of people with shared interests such as culture, politics, health, etc.

In this article, we present a review of literature solutions to the information retrieval problem in unstructured P2P networks and a description of the main techniques for routing queries in structured and semi-structured P2P systems. Our analysis includes a novel classification of these systems, putting special emphasis on their semantic aspects and the different existing routing algorithms. While existing algorithms have facilitated the implementation of robust distributed architectures, there are still several limitations faced by current search mechanisms. Indeed, the information retrieval problem is more complex than the traditional problem of searching for resources based on object identifiers or names. Over the years, the information retrieval community has developed numerous document retrieval techniques for centralized search. However, these methods cannot be directly applied to P2P information retrieval networks, where search is not centralized since documents are distributed among a large number of repositories. Given the information explosion that we have experienced in the last years, such new capabilities are an important step for making P2P networks effective in many applications that go beyond simple data storing. There has been previous research work which provided the background for our analysis: [147] presents a review of some early methods developed to address information retrieval problems in P2P networks. An empirical comparison of some of these methods is presented in [132] and a more recent survey of the major challenges for P2P information retrieval is presented in [129].

The remainder of the article is organized as follows: Section 2 presents some background concepts used in the rest of the paper, including a description of the main components of P2P networks and a classification of P2P search algorithms. Section 3 presents a summary of the most important search algorithms in P2P networks, highlighting those that make use of semantic aspects. Section 4 provides a comparison and classification of the algorithms presented in Section 3. Finally, the conclusions derived from this analysis are presented in Section 5.

2. Background

According to [117], a P2P network is, in its pure form, "a distributed system in which every peer communicates with other peers without the intervention of centralized hosts". In real-world P2P networks, the participating peers are typically computers to be found at the edge of the network, in people's offices or homes [77], Thus, a P2P network turns out to be formed by a set of machines, which offer a wide range of capabilities when considering storage and Internet access speeds, being attractive for different computing tasks (such as file sharing, media streaming, and distributed search). P2P networks have usually no centralized directory, being *self-organizing*, with the ability to adapt to different circumstances associated with the participating peers (e.g. joining in, failing or departing from the network). It is worth noticing that the use of a common language ensures that the communication between peers is *symmetric* for both the provision of services and communication capabilities. From this symmetry the P2P network can also be characterized as *self-scaling*, since each peer that joins the network adds a new computational resource

to the available total capacity [32,112,29]. There are many important challenges specific to P2P networks [45], such as how to administrate resources properly, how to provide an acceptable quality of service while guaranteeing robustness and availability of data, etc. Reviews of different P2P frameworks and their applications can be found in [24,105,89].

The rest of this section will present different dimensions relevant for assessing P2P networks. First, in subsection 2.1 we present a common classification for P2P networks. Then, subsection 2.2 introduces the concept of *semantic aspect* in the context of P2P information retrieval. Subsection 2.3 introduce a novel taxonomy for routing algorithms based on semantic aspects. Subsection 2.4 present the importance of distributed hash tables for the implementation of routing algorithms mainly for structured topologies. To conclude the section, subsection 2.5 introduces some of the salient applications of the P2P technologies.

2.1. Network classification

A common approach to classify P2P networks is based on their *degree of centralization*, which results in three possible alternatives:

- **Centralized**: These P2P networks have a monolithic architecture with a single server that allows transactions between nodes and keeps track of where content is stored. For example, *Napster* [100] had a constantly-updated directory hosted in a central location (the Napster website). This system was extremely successful before its legal issues [50]. Clearly, this centralized approach scales poorly and has a single point of failure.
- **Decentralized**: These are systems where there is no centralized directory, since each peer acts as a client and a server at the same time. *Gnutella* [111] is an example of this architecture, where the network is formed by nodes that join and leave the system. Several factors motivate the adoption of decentralized networks such as privacy control, availability, scalability, security, and reliability [107]. On the downside, to find a file in a decentralized network a node must query its neighbors. In its basic form, this method is called *Flooding* and is extremely unscalable, generating large loads on the network participants.
- **Hybrid**: These systems have no central directory server and therefore can be seen as decentralized networks. However, they have some special peers or super-peers with extra capabilities. *FreeNet* [41,4] is an example of such system and there is a growing interest on this kind of P2P architectures, which supports a hash-table-like interface [110,124,114,149,63].

The network topology is another common classification criterion for P2P networks, allowing to identify two distinctive groups:

- **Structured**: In structured P2P networks the nodes are organized into a specific topology. This organization ensures that any node can search the network for a resource, providing as well a good response time. The most common type of structured P2P network is based on a distributed hash table (DHT) that provides a lookup service similar to a hash table: (key, value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key. This approach is adopted by *Chord* [124], *Pastry* [114] and *Tapestry* [149], among others.

– **Unstructured**: In unstructured P2P networks no structure is pre-established over the network, but rather these networks are formed by nodes that randomly build connections to each other. Because of their lack of structure, unstructured networks are easy to build and highly robust. However, a major limitation of these networks is their poor effectiveness in finding resources. The simplest algorithm used on unstructured P2P networks is based on propagating the query message through all the network, leading to a high amount of traffic. Additionally, it is not possible to ensure that search queries will be eventually resolved. Some examples of this type of systems are *Gnutella* [111] and *KaZaa* [1].

Figure 2 shows a diagram with the classification of P2P systems, identifying the relationships between the different degrees of centralization and the possible topology of the network. Structured topologies are frequently related to centralized or hybrid systems in order to take full advantage of both features. However, unstructured topologies have random connections and in general any peer is equivalent to the others, so that they are strongly associated with the concept of decentralization. The items marked with a star correspond to the central topics on which we will focus on the rest of this survey.

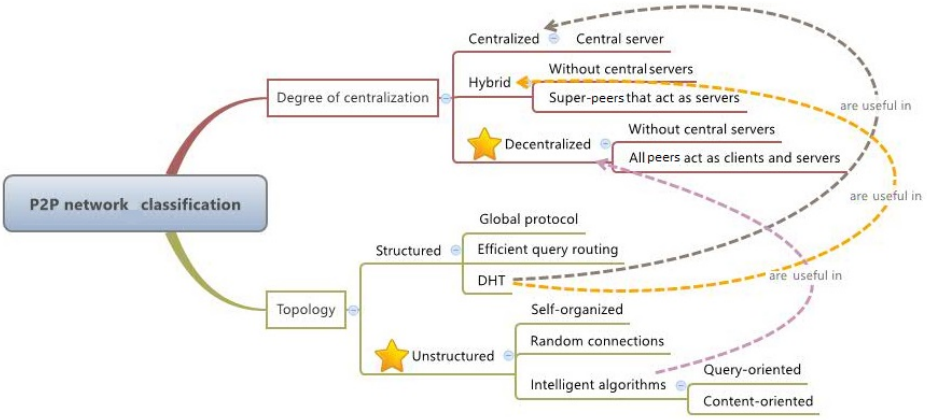


Fig. 2. Classification of P2P networks in terms of their degree of centralization and associated topology.

Search methods depend on the underlying network structure. As discussed before, in the case of pure unstructured networks there is no specific pattern for the organization of its nodes, resulting in a random topology. As a consequence, these networks have

relatively low search efficiency when contrasted with structured and hybrid ones, as a single query can generate massive amounts of traffic even if the network has a moderate size [112]. Thus, many alternatives have been studied to improve the basic flooding approach in unstructured networks, such as random walk [88,71,145,28,80], directing the search towards potentially useful nodes [21,147,138,120], or clustering peers by content [44,133,33] or interest [123,93,108].

2.2. Semantic Aspects

A possible solution to improve communication overhead and scalability in large-scale unstructured P2P systems is to forward queries to a group of peers that are known to be potentially useful to answer the query. The selection of potentially useful peers is typically based on the peers' past activity or their semantic similarity to the original query [43,123,135,74,85,78,130]. Identifying peers based on their potential to answer a query in a useful way requires associating semantic aspects with peers.

A *semantic aspect*, in the context of a P2P network, is a feature or a set of features that allows recognizing the semantic of the data stored in a node. Semantic aspects can be exploited by routing algorithms to help peers predict which other peers have knowledge useful to respond to a query. Topical information, past experience, and node-state information are examples of such semantic aspects.

In algorithms that use topical information to compute topic similarity, each peer stores profiles of other peers. A neighbor profile is information that a peer maintains to describe the content stored by a neighbor. By analyzing neighbor profiles, peers try to increase the probability of choosing appropriate peers to route queries. Algorithms that use topic similarity to guide the search process in a P2P network lead to the spontaneous formation of semantic communities through local peer interactions [22].

A key concept associated with the use of topic similarity to guide the search process is "semantic locality" [140]. Traditionally, the notion of semantic locality has been used to refer to the ability to store information about peers offering semantically close services. This ability can be used to index and locate content, complementing the current service discovery mechanisms in a Grid and in the Web [115,150,116,79]. Semantic locality has also been defined as "a logical semantic categorization of a group of peers sharing common data" [119]. With the help of locality information, an unstructured P2P network allows to design more informed mechanisms for routing queries, mitigating the complexity of the search process [102,103].

Another alternative for capturing semantic aspects consists in storing past experiences from the interaction of a peer with its neighbors. This approach does not require storing nodes' profiles. Instead, a peer keeps track of valuable routing information (such as the number of hits per node or peer availability) and uses this information to select the most active peers to forward a query [49]. The main problem with this approach is that it strongly benefits those peers that store large amounts of data, penalizing less resourceful peers that may also offer relevant material for specific topics.

A number of algorithms use heuristic information based on the state of the nodes and their past performance to select candidate nodes. There are many heuristics that can be considered; some of them are based on the analysis of the latency or the response time of specific nodes [151]. Clearly, in such cases, additional specific data must be collected and stored for computing the associated heuristic. For example, in [82] a heuristic-based query

routing algorithm is presented. This algorithm collects a plurality of metrics for each host that it is aware of in a P2P network, most often by host information or query hit messages. The metrics collected aid in determining a set of P2P hosts best able to fulfill a query message, without having knowledge of specific content. The metrics collected also aid in managing query messages in that they determine when to drop query messages or when to resend query messages. The choice of the heuristic is a very important step in the process of developing an algorithm. In [141] a study of the *DirectedBFS* algorithm implemented under different heuristics is presented. In this algorithm, in order to intelligently select neighbors, a node maintains simple statistics on its neighbors, such as the number of results received through that neighbor for past queries, or the latency of the connection with that neighbor. From these statistics, the authors develop a number of heuristics to select the best neighbor to send the query, such as:

- Select the neighbor that has returned the highest number of results for previous queries.
- Select the neighbor that returns response messages that have taken the lowest average number of hops. A low hop-count may suggest that this neighbor is particularly close to nodes containing useful data.
- Select the neighbor that has forwarded the largest number of messages (all types). A high message count would imply that this neighbor is stable and it can handle a large flow of messages.
- Select the neighbor with the shortest message queue. A long message queue implies that the neighbor's pipe is saturated, or that the neighbor has died.

In summary, the use of semantic aspects helps to select the most promising nodes to route queries with the purpose of implementing more informed search strategies.

2.3. Algorithm classification based on semantic aspects

In unstructured P2P networks, routing algorithms can be classified into *content-oriented* or *query-oriented*, based on the semantic aspects and the decision-making criteria used to route a query [26].

- **Content-oriented**: these routing algorithms use metadata extracted from the shared content of each peer to build a local index with global information. This index provides each peer with an approximate view of the network content and other peers' profiles. Hence, peers will be able to route efficiently their queries, improving the retrieval effectiveness. Nodes' profiles are the most used semantic aspects in content-oriented routing algorithms [43,76].
- **Query-oriented**: these routing algorithms exploit the historical information of past queries and query hits to route future queries. Past experience based on query hits is the most used semantic aspect in query-oriented routing algorithms [131,81].

Content-oriented algorithms produce a very large number of messages to build their associated indices. In contrast, query-oriented methods are more advantageous, as no excessive network overhead is required for building the routing indices. Recently, efficient approaches to content-oriented routing algorithms have been proposed in the context of content-centric networking [39,146]. In content-centric networking, a data object is retrieved based on its content identity instead of the IP address of the node on which it resides.

2.4. Distributed Hash Tables in P2P systems

Distributed Hash Tables (DHTs) are data structures for indexing data using a distributed approach. DHTs provide a powerful tool that has changed the way resources and information are shared. In structured P2P systems, the data objects are stored by a globally-agreed scheme. In this context DHTs have turned out to be one of the most highly used approaches [52]. In P2P systems implemented with DHTs, each peer represents a hash table bucket with a global hash function. Search in this kind of systems is guaranteed and efficient since it typically involves logarithmic time with respect to the overlay network size. A popular P2P system based on DHTs is Kademlia [91], which includes several desirable characteristics that were not present in previous DHT-based approaches. Kademlia minimizes the number of configuration messages that every node needs to send in order to learn about each other. In this system, each node has enough knowledge to be able to proceed with query routing using low-latency paths. Recent work on Kademlia has been oriented towards analyzing the resilience against failing nodes and communication channels [61] and secure and trustable distribution aggregation [57]. Viceroy [90] is another P2P system whose relevance lies on being the first P2P system to combine a constant degree with a logarithmic diameter, while still preserving fairness and minimizing congestion. This is achieved through a quite complex architecture that guarantees with high probability that the congestion of the network is within a logarithmic factor of the optimum. Later work on Viceroy resulted in Georoy [54], an algorithm for efficient retrieval of information based on the Viceroy P2P algorithm. Unlike Viceroy, Georoy establishes a direct mapping between the identification of a resource and the node which maintains information about its location. In spite of all their advantages, it must be remarked that systems based on DHTs suffer from limitations in terms of robustness and search flexibility. A good P2P structured system needs cooperation among peers to maintain flexibility and credibility. This assumption is particularly strong, as not all devices on the Internet connected through the network are necessarily stable and reliable.

2.5. Applications of P2P systems

Many software applications that gained popularity among a large community of users, such as *eMule* [14] and *PopCorn Time* [17], operate on P2P networks. In both cases, these applications use P2P technologies to stream audio and video to their end users, generating a considerable portion of the overall traffic on the Internet and requiring a large amount of energy consumption [34].

Regarding to educational settings, P2P technologies have allowed institutions to share files globally, as is the case of the *LionShare* project[96]. Another popular distributed application is *Bitcoin* and its alternatives, such as *Peercoin* and *Nxt*, which are P2P-based digital cryptocurrencies. *Bitcoin* [51] is a P2P system where transactions take place directly among users, without an intermediary. Network nodes are in charge of verifying these transactions, which are eventually recorded in a public distributed ledger (called the blockchain). *Bitcoin* has no central repository (nor administrator) and is known as the first decentralized digital currency [98].

Another area where P2P technologies are becoming increasingly important is social networking. Currently, the social web is mostly dominated by centralized social networks such as *Twitter*, *Facebook* and *MySpace* [18,15,16] and as a consequence it is limited by

the centralization in the use of the information and the potential loss of control of the privacy of the information by the users. These social networks create the illusion that users are directly connected to each other. However, the centralized servers belonging to companies are the ones in charge of controlling the data and the interactions among users. The lack of control of the users on their data is a problem that is aggravated by the terms of services, which are often unclear or have notable disadvantages, and by the occasional changes in the privacy policy. Distributed technologies offer a possible solution to the problem of centralization. With this approach, servers (peers) can communicate with each other without the intervention of a central server. This schema allows that data, and the control over them, to be distributed among all users and their servers. P2P networks offer a way to relax the constraints of centralized control, resulting in systems that are decentralized, concurrent and with collective or emerging behaviors. These features make P2P an attractive technology to support social networks. An example of open source distributed social network is *BuddyCloud* [2], which allows software developers to share their applications, supplemented with chats and videos. Another distributed social network is *Diaspora** [3], whose policy allows the decentralization in the use of information. In this social network profiles are stored in users' personal web servers, allowing them to have full control of the content they share and to have absolute knowledge of where the content is stored and who has access to it. Other examples of distributed social networks are *Friendica* [5], *GNU social* [6], *Mastodon* [8], *Minds* [9], *Kune* [7] and *Twister* [10].

Clustering is an important data mining issue, especially for large and distributed data analysis. Distributed computing environments such as P2P networks involve separated sources, distributed among the peers. According to unpredictable growth and dynamic nature of P2P networks, data of peers are constantly changing. Due to the high utilization of computing and communication resources and privacy concerns, processing of these types of data should be applied in a distributed way and without central management. In this scenario, clustering algorithms became important to organize the peers among the network. An example of this kind of algorithm is *GBDC-P2P* [27]. The *GBDC-P2P* algorithm is suitable for data clustering in unstructured P2P networks and it adapts to the dynamic conditions of these networks. In the *GBDC-P2P* algorithm, peers perform data clustering with a distributed approach only through communications with their neighbors.

Distributed data storage is another area in which P2P technologies have proven to be helpful. Distributed databases allow quick access to data stored throughout a network, and have different capabilities (e.g. some provide rich query abilities whereas others are restricted to a key-value store semantics). Google's Bigtable [12], Amazon's Dynamo [13], Windows Azure Storage [19], and Apache Cassandra [11] (formerly Facebook's data store [59]) are examples of distributed databases. In P2P network data storage, the user can usually reciprocate and allow other users to use their computer as a storage node as well. Information may or may not be accessible to other users depending on the design of the network.

3. Routing algorithms

In this section, we analyze around forty different algorithms as well as some of their variants related to intelligent query routing in P2P networks. In particular, as discussed previously, we will focus on unstructured P2P networks in which semantic issues play a

mayor role. We provide a brief description of each algorithm along with the corresponding references.

3.1. Routing algorithms in structured P2P networks

In a structured P2P system, the topology that defines the connections among peers and data locations is predefined. This pre-established topology is exploited by search mechanisms that take advantage of these pre-defined relations among peers. Even when using a distributed hash table (DHT), structured systems may differ on the data structures used for implementing it (e.g. some of them may rely on flat overlay structures while others might be based on hierarchical overlay structures). A benefit of DHTs is the possibility to exploit the structure of the overlay network for sending a message to all nodes, or a subset of nodes, ensuring a threshold for the overall execution time involved. It is not natural to implement a search algorithm with DHT in unstructured networks due to their lack of structure. However, some authors have explored their application in unstructured and semistructured networks [62].

Some flat data structures (Figure 3) include ring, mesh, hypercube, and special graphs such as the de Bruijn graph [46]. For example, *Chord* [124] uses a ring data structure with node IDs. Each node keeps a table that contains the IP addresses of those nodes that are half of the ID ring away from it. A key k is mapped to a node A whose ID is the biggest that does not exceed k. In the search process, A forwards the query for key k to $succ(k)$ (node in A's table with the highest ID that is not larger than k). In this way, a query can be forwarded until the node that holds the key is reached. The so-called "finger table" speeds up the lookup operation, ensuring an execution time of $O(logN)$.

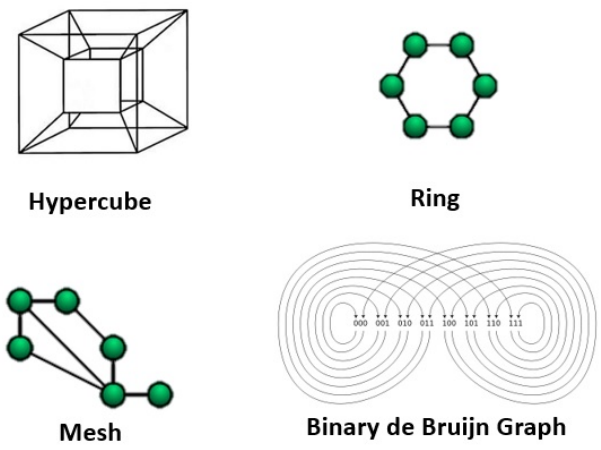


Fig. 3. Some examples of possible structured topologies.

Pastry [114] is based on a tree data structure which can be considered a generalization of a hypercube. Each node A keeps a leaf set L. For every node A, the set L consists of

those $L/2$ nodes whose IDs are nearest to and smaller than the ID of A, along with the set of $L/2$ nodes whose IDs are nearest to and bigger than the ID of A. This set ensures the correctness of the process. Every Pastry node also keeps a routing table of references to other nodes of the same ID space. In Pastry, given a search query q associated with a key k, a node A forwards q to a node whose ID is the nearest to k among all the nodes known to A. Node A tries then to find a node in its leaf set. If that node does not exist, A looks for a candidate node in its routing table whose ID shares a longer prefix with k than A. If this node does not exist either, A forwards q to a node whose ID has the same shared prefix than A but is numerically closer to k than A. In this way, each Pastry node ensures an execution time of $O(logN)$. The approach presented in [149] called *Tapestry* is similar to the previous one. They differ in the underlying routing algorithm and in the approach taken to exploit the locality. Tapestry also ensures an execution time of $O(logN)$.

In *CAN* [110] DHTs are implemented using a d-dimensional toroidal space divided into hyper-rectangles, which define different zones. Each of these zones is controlled by a particular node. Keys are mapped with a hash function to points in the d-dimensional space. Each node has a routing table that consists of all of the other nodes that are in its d-dimensional space. A node A is in the same space of another node B if the zone of B shares a $(d-1)$-dimensional hyperplane with the zone of A. Given a query q associated with a key k, a node forwards q to another node according to its routing table whose zone is the nearest to the zone of the node responsible for key k.

The *NetSize-aware* protocol introduced in [148] is based on *CAN*. The main objective of this algorithm is to solve the problem of search flexibility in DHT. This algorithm preserves CAN's simplicity, providing a greedy routing algorithm based on DHT. *NetSize-aware* uses a binary partition tree algorithm to determine the underlying network topology. Simulation results show that this approach is resilient, efficient and improves the performance of *CAN*.

KaZaA [1] is a P2P file-sharing technology that was commonly used to exchange MP3 music files and other file types (such as videos, applications, and documents) over the Internet. Its architecture is based on a two-tier hierarchy in which some nodes are distinguished as *supernodes*. Supernodes are those nodes with the fastest Internet connection and best CPU power. Each supernode is responsible for indexing the files of the nodes that it handles. The use of supernodes with better computing capabilities than regular nodes allows the system to perform better than the local-indices approach respect to lower susceptibility to bottlenecks, and similar resilience to churn (where the churn rate can be defined as a measure of the number of individuals or items moving out of a collective group over a specific period of time). However, this system suffers the problem of the resulting overhead associated with exchanging index information between regular nodes and supernodes [143,84]. In [65] a routing algorithm that is also structured in two layers is proposed: *SkipNet* layer and *Small-World* layer. The first layer routes the queries based on a numerical ID and the second layer routes the queries using a Small-World topology (see [137] for a pioneering study of Small-World networks).

An efficient P2P information retrieval system called *pSearch* is presented in [127]. *pSearch* supports semantic-based full text searches and avoids the scalability problem of certain systems that employ centralized indexing. In *pSearch* documents are organized based on their vector representations generated by information retrieval algorithms based on the vector space model and latent semantic indexing. This organization results in more

efficiency and accuracy, as the search space for a particular query is defined on the basis of related documents.

The growth of intercommunication between computers gives systems the chance to operate more efficiently, by better supporting the cooperation between individual components. *AFT* [106] is an overlay that adapts to a changing number of nodes in a P2P network and is resilient to faults. The *AFT* overlay is designed to be a solution for systems that need to share transient information, performing a synchronization between various components, such as in mobile ad-hoc networks, urban networks, and wireless sensor networks. The operations supported by the overlay, such as joining, leaving, unicast transmission, broadcast sharing and maintenance can be accomplished in time complexity of $\mathcal{O}(\sqrt{N})$, where N is the number of nodes which are part of the structure.

In [146] a novel framework is introduced, based on implementing a hybrid forwarding mechanism. This approach allows discovering content in a proactive or reactive way based on content characteristics. The proposed framework classifies time-sensitive data utilizing content identifiability and content name prefixes, aiming at applying the most suitable strategy to each category. For proactive content dissemination they propose a Hierarchical Bloom-Filter based Routing algorithm (see [35] for a detailed review of the concept of bloom filters). A Hierarchical Bloom-Filter is structured in a self-organized geographical hierarchy, which makes the approach scalable to large metropolitan Vehicular Ad-Hoc Networks (VANETs).

An approach presented in [79] characterizes the notion of semantic-based sub-spaces as a basis for organizing the huge search space of large-scale networks. Each sub-space consists of a set of participants that share similar interests, resulting in semantic-based Virtual Organizations (VOs). Thus the search process occurs within VOs where queries can be propagated to the appropriate members. The authors propose a generic ontological model that guides users in determining the desired ontological properties and in choosing the "right" VOs to join. DHTs are used to index and lookup the hierarchical taxonomy in order to implement the ontology directory in a decentralized manner. Even though the ontology-based model facilitates the formation of the VOs, searching and sharing efficiently is still a major challenge due to the dynamic and large-scale properties of the search space. In order to efficiently share and discover resources inside VOs an infrastructure called *OntoSum* is proposed.

Security is an important feature in all type of networks, especially in P2P networks where every participant requests and provides information without any centralized control. To prevent structured overlay networks from being attacked by malicious nodes, a symmetric lookup-based routing algorithm referred to as *Symmetric-Chord* is presented in [87]. This algorithm determines the precision of routing lookups by constructing multiple paths to the destination. The selective routing algorithm is used to acquire information on the neighbors of the root. The authenticity of the root is validated via consistency shown between the information ascertained from the neighbors and information from the yet-to-be-verified root, resulting in greater efficiency of resource lookup. Simulation results demonstrate that *Symmetric-Chord* has the capability of detecting malicious nodes both accurately and efficiently, so as to identify which root holds the correct key, and provides an effective approach to the routing security for the P2P overlay network.

Another approach that implements an attack detection method is presented in [66]. The authors propose a routing table "sanitizing" approach that is independent of a spe-

cific attack variant. The proposed method continuously detects and subsequently removes malicious routing information based on distributed quorum decisions, and efficiently forwards malicious information findings to other peers which allows for progressive global sanitizing.

In [23], the authors have proposed a scalable solution for lookup acceleration and optimization based on the de Bruijn graph with right shift. The proposed solution is principally based on the determination and elimination of the common string between source and destination. This procedure is executed locally at the current requestor node. The performance aspects of the proposed model have been validated through simulation results developing a specific Java program. Among other approaches that use de Bruijn graphs we can cite D2B [53], DH-DHT [99] and Koorde [70].

3.2. Routing algorithms in semi-structured or loosely structured P2P networks

In loosely structured P2P networks the overlay structure is not strictly specified. The emerging structure turns out to be formed in a probabilistic way, or defined by some underlying topology. Thus, searching in this kind of networks depends on the overlay structure and how the data is stored [125]. *FreeNet* [41,4] is a P2P loosely structured system designed for protecting the anonymity of data sources. This scheme is based on the DHT interface, where each node has a local data repository and an adaptable routing table. These tables have information about addresses of other nodes and the possible keys stored in these nodes. Searches are performed in the following way: let us assume that node A is the query-issuing node and it generates a query q for a data item with key k. First, A looks up its data repository. If the file is found locally, q is resolved. Otherwise, q is forwarded to the node B whose key is nearest to k according to As routing table. Then, node B performs a similar computation. This procedure continues until the search process terminates. During this process, a node may not forward the query to the nearest-key neighbor because that neighbor is down or a loop is detected. In such cases, this node tries to contact the neighbor with the second nearest key. A TTL (time to live) limit is specified to restrict the number of messages in the query routing process. If the data item is found, the file is returned to the query-issuing node in the reverse path of the query. Each node (except the last one on the query path), creates an entry in the routing table for the key k. To bring anonymity, each node can change the reply message and claim itself or another node as the data source.

PHIRST [113] is a system that aims at facilitating full-text search within P2P databases and simultaneously takes advantage of structured and unstructured approaches. In a similar way to structured approaches, peers publish first terms within their document space. The main difference with respect to other algorithms is that frequent terms can be quickly identified and do not need to be stored exhaustively, thus reducing the storage requirements of the system. In contrast, during query lookup agents use unstructured search to compensate for the lack of fully published terms. In this way the costs of structured and unstructured approaches are balanced, achieving a reduction in the costs involved in the queries that are generated in the system. There are other kinds of semi-structured P2P networks where the network is divided into different subnets, resulting in a topology based on the peers' interests. In [93] a system is presented where nodes are clustered according to their interests [33] to form a P2P overlay network of multilayer interest domains. Three

types of nodes are distinguished: active nodes, super-nodes, and normal nodes. Each active node acts as a router providing information that facilitates query routing information at the cluster level. Each super-node is responsible for maintaining the related information of each member of the cluster. Finally, normal nodes are responsible for providing and sharing resources. The network resulting topology is shown in Figure 4.

There are other kinds of semi-structured P2P networks where the network is divided into different subnets, resulting in a topology based on the peers' interests. In [93] a system is presented where nodes are clustered according to their interests [33] to form a P2P overlay network of multilayer interest domains. In this system, there are three types of nodes: active nodes, super-nodes, and normal nodes. Each active node acts as a router providing information that facilitates query routing information at the cluster level. A super-node is a representative node of a cluster and is in charge of maintaining the related information of each member node in the cluster. Finally, a normal node is mainly responsible for providing various types of shared resources. The resulting network topology is shown in Figure 4. Another similar approach is presented in [133], where the authors propose a system in which clusters are constructed on multiple logical layers. In this system, peers can switch overlay networks to search content based on popularity. One of the overlay networks is a network based on clusters constructed according to the content of each peer [134,104].

Social media has changed our way of communication and sharing data on the Internet, which is now mostly based on collaboration among members to provide and exchange information. The efficiency of this new form of interaction motivates researchers to design architectures based on the social behavior of the users. In [30] an algorithm called *ROUTIL*, that combines social computing and P2P systems, is presented. The goal in ROUTIL is to link users with similar interests in order to provide a secure and effective service. A method for modeling users' interest in P2P-document-sharing systems based on k-medoids clustering is presented in [108]. In the proposed approach an overlay network is created based on the k-medoids clustering algorithm, which is combined with the users' historical queries to improve the initial user interest model.

3.3. Routing algorithms in unstructured P2P networks

In an unstructured P2P network (Figure 5), there is no specific criterion which strictly defines where data is stored and which nodes are neighbors of each other. The *Breadth First Search* (BFS) or flooding is the typical algorithm used to search in pure P2P networks. In these algorithms, queries are propagated from a node to all of its neighbors, then to the neighbors of those nodes and so on, until the TTL parameter becomes zero. This routing method is implemented in some systems such as Napster and Gnutella [100,111]. Flooding tries to find the maximum number of results. However, flooding does not scale well [111], and it generates a large number of messages in comparison with other approaches.

Many alternative schemes have been proposed to address the original flooding problems in unstructured P2P networks. In *iterative deepening* [142], also called expanding ring, the query-issuing node periodically carries out a sequence of BFS searches with increasing depth limits $D_1 < D_2 <...< D_n$. The query is considered to be resolved when the query result is satisfied or when the maximum depth limit *n* has been reached. In the latter case, the query is assumed to remain unsolved, and it can be determined that the query-issuing node will never find the answer to that query. All nodes use the

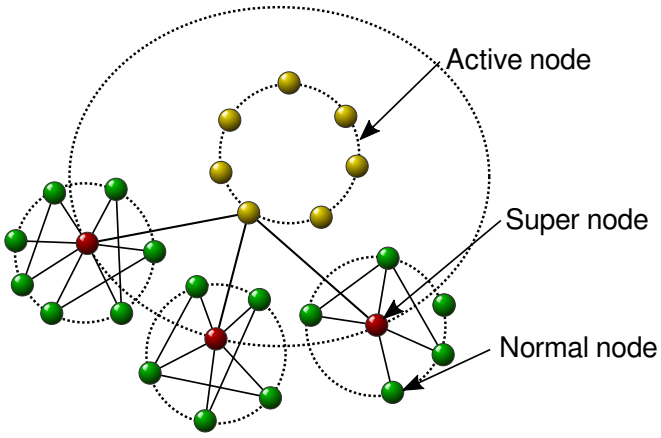


Fig. 4. Network topology structure in a P2P overlay network of multilayer interest domains (adapted from [93]).

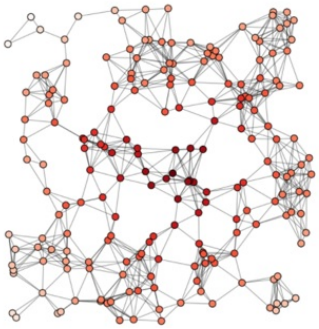


Fig. 5. Graphical representation of an unstructured topology in a P2P network.

same sequence of depth limits called *policy P* and the same period of time between two consecutive BFS searches. This algorithm is appropriate for applications where the initial number of query hits is important, but this approach does not reduce the number of duplicate messages and the associated query processing time is high.

In a *Depth-First Search* (DFS) algorithm, rather than sending a query to all the neighbors, each peer selects a single candidate neighbor to send the query. In this scheme, the maximum TTL of a query is used to specify the search depth. If the query-originating node does not receive a reply within a certain period of time, the node selects another neighbor to send the query. The process is repeated until the query is answered or all the neighbors have been selected. The criteria used to select a neighbor can highly influence the performance of the search process. FreeNet [41,4] is an example of a P2P system using the DFS scheme.

In the standard *random-walk* algorithm [55], the query-issuing node forwards the query to one neighbor selected randomly. On its turn, this neighbor proceeds in a similary way, choosing randomly one of its neighbors and forwarding the query message to that neighbor. The procedure is repeated until the required data is found. This algorithm uses only one walker, reducing the message overhead but causing longer search delays. In the *k-walker random walk* [88], k copies of the query message are sent by the query-issuing node to k randomly selected neighbors. Each query message takes its own random walk. In order to decide if a termination condition has been reached, each walker periodically communicates with the query-issuing node. This algorithm attempts to reduce the routing delay. A similar approach is the *two-level random-walk* algorithm [67]. In this algorithm, the query-issuing node uses k_1 random search threads with a TTL with a value of l_1. When this TTL parameter expires, each search thread explodes to k_2 search threads with the TTL parameter established in l_2. This approach aims to reduce duplicate messages, but it has a longer search delay than the k-walker random walk. Random-walk approaches are popular in P2P applications. For example, in [80] a study of the random-walk domination problem is presented with the formulation of an effective greedy algorithm that guarantees an optimal performance.

Another similar approach is the *modified random BFS* algorithm [71] where the query-issuing node forwards the query to a randomly selected subset of its neighbors. On receiving a query message, each neighbor forwards the query to a randomly selected subset of its neighbors (excluding the query-issuing node). This algorithm continues until some stop condition is satisfied. As pointed out in [71], this approach results in more nodes being visited and has a higher query success rate than the k-walker random walk.

Directed BFS [142] is a routing algorithm that selects those neighbors from the query-issuing node which are expected to quickly return many high-quality results. The selected neighbors subsequently forward the query message in a BFS way to all their neighbors. Each peer stores simple statistics about its neighbors (e.g. the highest number of query results returned previously, network latency for the neighbor, or the least busy neighbors) and uses this information for a more informed neighbor selection strategy.

Intelligent search [71] is similar to directed BFS. However, a more intelligent approach to neighbor selection is achieved by considering the past performance of the query-issuing node neighbors and limiting query propagation only to a selected subset of these neighbors. These neighbors are selected through a query-oriented approach that considers whether the neighbors have successfully answered similar queries (based on query cosine

similarity [64]) in the past. Each node keeps a profile of its neighbors with information on those queries that the neighbors answered more recently in the past. Similarly to other query propagation approaches, a TTL value is used to stop the query propagation process.

In the *local-index-based search* algorithm [142], every node replicates the indices maintained by other nodes for their local data with a k-hop distance from it. In this way, a node can use data from its local indices to answer queries associated with data stored in other nodes. A broadcast policy P defines when query propagation must stop. As a consequence, only those nodes at depths smaller than those listed in P check their local indices and return the query result if the requested data is found. Local indices are updated to reflect changes when a node joins, leaves, or modifies its own data. At the time a node Y joins the network it sends a join message with a TTL of r hops. Hence, all nodes within an r-hop distance from Y receive this message. The message contains metadata describing Y's data collection. If a node X receives a join message from Y, it replies with another join message with metadata describing its own data collection to keep Y's index up to date. Each time a node Z leaves the network or dies, other nodes that index Z update their indices after a timeout, removing information on Z's data collections from their indices. Modifications on Z's data collections are reflected on other nodes indices by sending a short update message with a TTL of r to all Z's neighbors. Query propagation in the local-index-based search approach is similar to the iterative deepening approach in that both algorithms rely on a list of depths to limit the number of hops allowed. However, while in iterative deepening nodes maintain indices containing local information only, in local-index-based search, nodes maintain indices containing not only local information but also information about data collections from other nodes.

The *routing-index-based search* algorithm [43] is similar to directed BFS and intelligent search. The three approaches guide the entire search process using neighbor information but differ in the type of information stored and the way this information is used. In directed BFS only the query-issuing node uses this information to select appropriate peers while the rest of the nodes use BFS as a strategy to route queries. Intelligent search uses information about the past queries that have been answered by neighbors. However, different from the rest, routing-index-based search stores information about the number of documents and the topics of the documents stored in the neighbor nodes. This facilitates the process of selecting the best candidate neighbors to forward queries. In routing-index-based search, good neighbors typically provide a means to quickly find many documents. Since indices are required to be small in a distributed-index mechanism routing indices do not maintain the location of each document. Instead they maintain information to guide the process of finding a document.

To illustrate the routing indices approach, we revisit the example presented in [43]. Consider Figure 6 which shows four nodes A, B, C, and D, connected by solid lines. The document with content x is located at node C, but the RI of node A points to neighbor B instead of pointing directly to C (dotted arrow). By using "routes" rather than destinations, the indices are proportional to the number of neighbors, rather than to the number of documents. The size of the RIs is reduced by using approximate indices, i.e., by allowing RIs to give a hint (rather than a definite answer) about the location of a document. For example, in the same figure, an entry in the RI of node A may cover documents with contents x, y or z. A request for documents with content x will yield a correct hint, but one for content y or z will not. This is a content-oriented method such that the node

knowledge about topics belonging to other peers is updated when a node establishes a new connection (not by past experience).

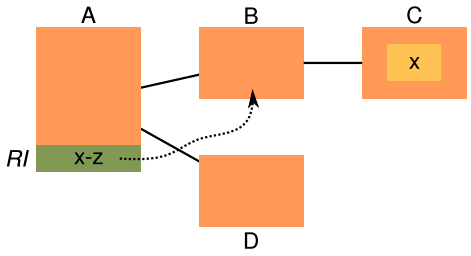


Fig. 6. Routing Indices schema (adapted from [43]).

Some P2P information retrieval methods are adaptations of a classification problem to query routing. In a classification problem, the classifier tries to classify an object using some features. The *Semantic Overlay Model* [68] aims at locating appropriate peers to answer a specific query. Instead of broadcasting queries, this approach routes queries to semantically similar peers. They produce semantic vectors in order to classify peers into categories that represent the peers' semantic similarity. This is a content-oriented method given that it uses meta-information to classify peers by interests. A query can be routed to related peers, increasing the recall rate while reducing at the same time hops and messages. Experiments have shown that establishing a semantic overlay model based on latent semantic indexing and support vector machine methods is feasible and performs well and that the query routing algorithm in the semantic overlay is efficient [68] .

In a pure P2P unstructured and decentralized network all the peers usually have the same responsibilities. However, some query-routing methods in unstructured P2P network make use the notion of "super-peer" as is the case in the *Backpressure* algorithm [118]. In this algorithm, super-peers serve their subordinates by resolving queries or forwarding them to other super-peers. Super-peers can resolve queries by checking the files/resources they have, as well as those of their subordinate community. Methods that impose some structure on special peers are considered "routing algorithms for semi-structured P2P networks", but in this case super-peers are self-organized without a central or initial control. The algorithm *Backpressure* is query-oriented, and it uses past information to decide how to route queries disregarding the content of each peer.

The *Route Learning* algorithm [40] uses keywords extracted from queries to determine how to route the queries. This differs from other approaches, where meta-data is used to classify queries and to decide how to route them. In the Route Learning scheme, a peer tries to estimate the most likely neighbors to reply to queries. Peers calculate this estimation based on the knowledge that is gradually built from query and query hit messages sent to and received from the neighbors. Route Learning reduces the query overhead in flooding-based networks using keywords extracted from queries, being therefore a query-oriented method.

Routing future queries using past experiences is the best way to route queries to specific nodes with the objective of improving performance, but it is important to store this

accumulated knowledge in an efficient way. As a consequence, each peer may need to consider some storage space for maintaining metadata. This storage also implies the cost of keeping track of these data updates. The *Learning Peer Selection* approach [26] implements a query-oriented method on unstructured networks with the ability to discover users' preferences by analyzing their download history. The proposed model is implemented in three layers. The first of these layers is especially dedicated to store and to update past information. The other two layers are responsible for managing users' profiles and selecting the relevant peers to send queries.

Cooperation among peers in a P2P network is strongly linked with the concept of knowledge sharing. Usually, there is a trade-off between improving the network global knowledge and the cost of sending update messages through the network. The *Self Learning Query Routing* algorithm [38] attempts to improve global knowledge by learning the nodes' interests based on their past search result history. The number of shared files determines a rank of friendship between two nodes. Queries are initially routed to friend nodes only. In case of failure, a broadcast search is executed. Past search results allow nodes to incrementally learn about other nodes in the network that share the same interests.

A P2P algorithm that relies on the notion of semantic communities is *INGA* [83]. The INGA algorithm assumes that each peer plays a different role in a social network , such as content provide, recommender, etc. The roles associated with peers allow INGA to determine the best matching candidates to which a query should be forwarded. Facts are stored and managed locally on each peer, constituting the *topical knowledge* of the peer. Each peer maintains a personal semantic shortcut index. An evaluation of different P2P search strategies based on the *6S* system [139] is carried out in [22] with the purpose of showing the emergence of semantic communities. The query-routing evaluated strategies include a *random*, a *greedy* and a *reinforcement learning* algorithm. To route queries appropriately, in the greedy and the reinforcement learning algorithms, each peer learns and stores profiles of other peers. A neighbor profile is defined by the information that a peer maintains in order to describe the contents stored by a given neighbor. By adapting the profile information, peers try to increase the probability of choosing the appropriate neighbors for their queries. Simulations demonstrate that peers can learn from their interactions to form semantic communities even when the overlay network is unstructured. Another content-oriented search algorithm is *State-based search* (SBS) [138]. In this algorithm, each node maintains a list with state information associated with the other nodes in the network and uses this information to route queries. Searches are performed using a local fuzzy logic-based routing algorithm. Results reported by the authors indicate that SBS reduces the response time and obtains a better load balance when compared to baseline algorithms.

An approach aimed at achieving low bandwidth is *Scalable Query Routing* (SQR) [76]. In this algorithm, a routing table is maintained at each node that suggests the location of objects in the network based on the past experience. A data structure called *Exponentially Decaying Bloom Filter* (EDBF) encodes probabilistic routing tables in a highly compressed manner and allows efficient query propagation. Other content-oriented methods seek to control the system congestion by tracking alternative routes to balance the query load between peers. For instance, the method presented in [120] relies on a *Collaborative Q-Learning* algorithm that learns several parameters associated with the network state and performance. In [31] two algorithms are presented that are a combination of other existing techniques. One algorithm is a combination of *Flooding* and *Random Walk*

while the other combines *Flooding* with *Random Walk* with Neighbors Table. The authors present different results obtained from simulations over an unstructured P2P network that showed that hybrid algorithms provide the most balanced performance regarding the average number of hops, average search time and the number of failures when compared to the basic resource discovery algorithms.

Other relevant search algorithms in unstructured P2P networks that are not described in this article but are classified in the following section are *q-pilot* [126], *SemAnt* [95], *Remindin'* [128], *P2PSLN* [152] and *NeuroGrid* [69].

4. Comparative Analysis

Next we will present a comparative analysis of the major features involved in the query routing process. We will also discuss the advantages and disadvantages of the different existing approaches.

4.1. Features comparison

In this subsection, a comparative analysis of the algorithms previously described is presented. Table 1 shows a comparison between routing algorithms in structured P2P networks. In this kind of systems, the use of DHT allows ensuring a logarithmic execution time. The algorithm used by the *SkipNet* and *Small-Word* scheme shows a central difference with respect to the other algorithms presented in table 1. In Chord and Pastry, the goal is to implement a DHT diffusing content randomly throughout an overlay in order to obtain a uniform and load-balanced behavior, whereas in *SkipNet* the goal is to enable systems to preserve useful content and path locality using the Small-World topology to take advantage of shortcuts to remote nodes.

Table 1. Comparison of salient features that characterize structured P2P networks.

		Features			
		DHT	Overlay		Structure
			Flat	Hierarchical	
Algorithm	Chord	•	•		Ring
	Pastry	•	•		Tree
	Tapestry	•	•		Tree
	CAN	•	•		Toroidal
	KaZaA	•		•	2-layers
	SkipNet and Small World			•	2-layers
	pSearch	•		•	Toroidal
	AFT		•		Toroidal
	HBFR			•	Geographical
	OntoSum	•		•	Tree

In unstructured P2P networks, search turns out to be a difficult, non-scalable process [75]. As a consequence, these algorithms take advantage of different semantic as-

pects in order to optimize their associated search processes. Table 2 outlines the semantic aspects that are present in each of the unstructured systems described above. The first five are flooding-like algorithms, and consequently they do not consider any semantic aspect. In the rest of the algorithms, the goal is to strategically select candidate nodes in order to reduce query propagation. To do that, some algorithms (e.g. Directed BFS) use heuristic information, while others select the candidate nodes by past experience (query-oriented) or by analyzing the profile of a node (content-oriented). Finally, there is a subset of algorithms that use a classifier to decide which are the best candidate nodes.

Table 2. Semantic aspects in unstructured P2P networks.

		Semantic Aspects			
		Heuristic Information	Content Oriented	Query Oriented	Classification
Algorithm	Flooding				
	Iterative Deeping				
	Random Walk				
	K-walker Random Walk				
	Two-level K-walker Random Walk				
	Modified Random BFS				
	Directed BFS	•		•	
	Intelligent Search			•	
	Local Indices Based Search	•		•	
	Routing Indices Based Search		•		
	Semantic Overlay Model		•		•
	Route Learning			•	•
	Learning Peer Selection			•	
	Self Learning Query Routing			•	
	6S - Random				
	6S - Greedy		•		
	6S - Reinforcement Learning		•		
	q-pilot				•
	SemAnt	•			•
	REMINDIN'			•	
	P2PSLN		•		
	NeuroGrid		•		
	INGA		•		
	SQR	•			
	SBS		•		
	Collaborative Q-Learning	•	•		

There are some algorithms (such as BFS, DFS, and random approaches) that do not exploit semantic aspects and consequently they are forced to implement a less informed method for propagating queries. These features can be observed in table 3. From this table, we can appreciate that even those algorithms that account for semantic aspects have

a basic mechanism to propagate queries. These mechanisms are executed over the subset of candidate nodes or when no candidate node exists and queries must be propagated in an alternative way. Another feature that is present in this kind of algorithms is the TTL parameter, which is decremented by one every time the message goes from a node to another. By performing this process, when the value of the TTL parameter becomes zero the search for candidates can be assumed to be over, so that the original message can be ultimately discarded.

Table 3. Basic routing algorithms in unstructured P2P networks.

		Features			
		BFS	DFS	Random	TTL
Algorithm	Flooding	•			•
	Iterative Deeping	•			•
	Random Walk	•		•	•
	K-walker Random Walk	•		•	
	Two-level K-walker Random Walk	•		•	•
	Modified Random BFS	•		•	n/a
	Directed BFS	•			n/a
	Intelligent Search	•			•
	Local Indices Based Search	•			•
	Routing Indices Based Search		•		n/a
	Semantic Overlay Model	n/a	n/a	n/a	•
	Route Learning	n/a	n/a	n/a	•
	Learning Peer Selection	n/a	n/a	n/a	•
	Self Learning Query Routing	•			n/a
	6S - Random	•		•	•
	6S - Greedy	•			•
	6S - Reinforcement Learning	•			•
	q-pilot	n/a	n/a	n/a	•
	SemAnt	n/a	n/a	n/a	•
	REMINDIN'	n/a	n/a	n/a	n/a
	P2PSLN	n/a	n/a	n/a	•
	NeuroGrid	•			•
	INGA	n/a	n/a	n/a	n/a
	SQR	n/a	n/a	n/a	n/a
	SBS	n/a	n/a		•
	Collaborative Q-Learning	•			•

Figure 7 presents a timeline that shows the evolution of the main routing algorithms in P2P networks over the years. From this timeline we can see that between 2002 and 2005 there was a considerable growth of algorithms for unstructured networks, whereas in recent years research efforts have been particularly focused on hybrid and structured networks. Furthermore, it can be seen that among algorithms for searching in unstructured

networks, intelligent algorithms are still a minority, being most of them based on basic flooding mechanisms.

As introduced in Section 2, query routing algorithms can be classified according to the topology of the underlying network. Algorithms for structured networks use some data structure in order to select destination peers. Most of these algorithms use data structures such as trees or rings, but there are other less usual structures such as the geographical position of the peers or toroidal. Algorithms for query routing in unstructured P2P networks can be classified according to the degree of intelligence that they use to select destination peers. Most of these algorithms are based on basic routing techniques using a random parameter or simply based on graph paths. Intelligent algorithms use different strategies for routing queries, which range from the adoption of particular classification techniques to the use of different kinds of heuristics. Figure 8 shows the main features present in structured/unstructured routing algorithms.

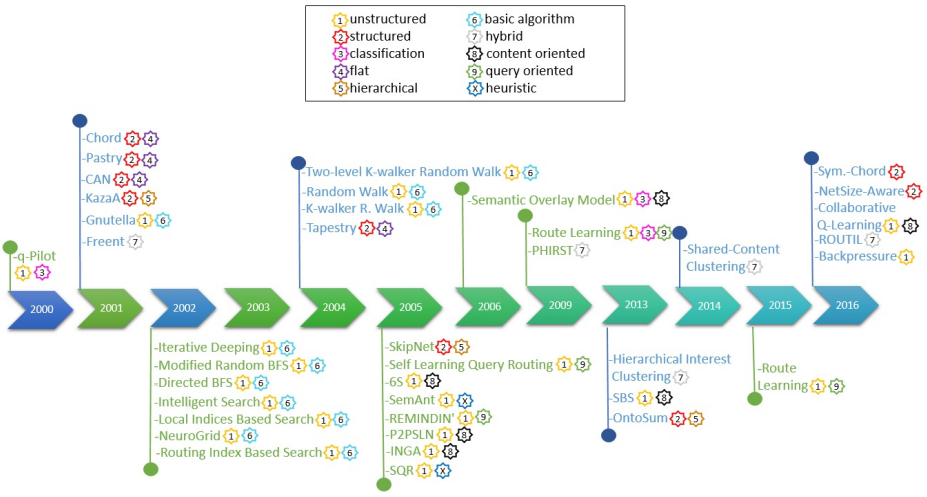


Fig. 7. Evolution of the main routing algorithms.

4.2. Discussion

P2P systems are distributed systems consisting of interconnected nodes that offer support to different applications such file sharing, distributed data storage, and distributed social networks, among others. Developing reliable, robust, effective and efficient P2P systems gives rise to research challenges such as the design of reputation systems in P2P

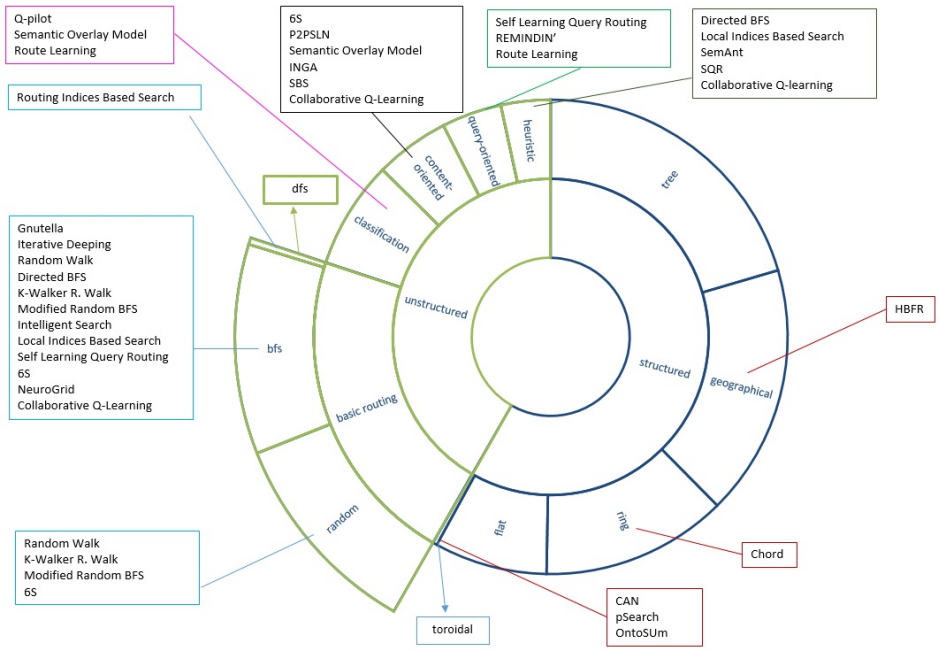


Fig. 8. Salient features of structured/unstructured routing algorithms .

environments, the exploitation of the semantic organization of information, the use of cryptographic mechanisms for data protection, etc. Several design features commonly considered when developing P2P systems include:

- **Replication.** P2P systems rely on content replication to ensure content availability. Replication is a major challenge for structured systems such as *Chord*, where identifiers are linked to their location. In these cases alias are used to allow replication [124]. *CAN* utilizes a replica function to produce random keys to store copies at different locations [136].
- **Security.** The dynamic and autonomous nature of P2P systems poses several challenges at the moment of ensuring availability, privacy, confidentiality, integrity, and authenticity [25]. Security in P2P networks is a highly explored field. Several cryptography algorithms and protocols have been developed especially for P2P systems, such as *Self-Certifying Data* and *Information Dispersal* [37,109]. Security also involves detecting and managing malicious nodes that can corrupt messages that are propagated among other nodes. The "Sybil Attack" [47] is a security threat related to authenticity where a node in a network claims multiple identities. The *Symmetric Chord* routing algorithm addresses the authenticity problem by implementing an authenticity validation process. Other approaches to address security issues in P2P networks rely on the use of special forms of access control lists [45].
- **Anonymity.** Author anonymity or peer anonymity is some times required in P2P applications. An approach named "Disassociation of Content Source and Requestor"

adopted by *FreeNet* provides anonymity to users by preventing other nodes from discovering the true origin of a file in the network. Another anonymity mechanism is "Censorship Resistant Lookup", which is used by *Achord* [60], a variant of the *Chord* lookup service.

- **Incentive mechanisms.** The performance of a decentralized P2P system relies on the voluntary participation of its users. To achieve a good performance it is necessary to implement methods that provide incentives to stimulate cooperation among users [58]. A simple incentive mechanism is based on ranking highly the results of a particular node if it has contributed significantly in previous searches. This simple method typically works for both the Web and P2P networks since appearing high up in a ranking typically represents an incentive for companies and people [42].
- **Semantic grouping of information.** The semantic organization of content through the emergence of semantic communities is deeply analyzed in [22]. Some systems are based on the notion of "peer communities", where relationships among peers are based on nodes' interests [73]. The emergence of these communities tends to make the search process more effective as it is possible to target search queries to those communities more closely related to the topic of the queries. This feature is exploited by some routing algorithms as discussed in [22,83].

There are several systems that use P2P technologies, such as those presented in section 2.5. These systems adopted different architectures and routing algorithms. We consider that no architecture is better than another, but can be use in deferents contexts. While a structured architecture can guarantee a determined execution time, a decentralized one can adapt more easily to topology changes. Decentralized approaches need to store some data to decide how to route a query but most of the algorithms for structured topologies need to keep their DHT update. Finally, as P2P technologies are still evolving, there are some open research problems such as a) developing routing algorithms for maximizing performance; b) defining more efficient security, anonymity, and censorship resistance schemes; c) exploiting semantic grouping of information in P2P networks; d) developing more effective incentive mechanisms.

5. Conclusions

The early Internet was designed on principles of cooperation and good engineering. In many ways, it shared principles and concepts with pure P2P networks. In this decentralized scenario, algorithms that performed searches were essential. When Internet became more rigid, structured and semi-structured search algorithms emerged, where collaboration among peers was no longer an important issue. Nevertheless, in the last few years pure P2P networks have come back for playing a major role in the deployment of new systems and technologies. Research in decentralized search has given rise to novel algorithms that incorporate semantic aspects derived from the profile of each participant. These semantic aspects can be conveniently exploited to improve routing algorithms, with the goal of minimizing network traffic and optimizing query response time.

In this article, we have reviewed the most important query routing algorithms in P2P networks, contrasting their advantages and disadvantages. To facilitate the analysis of these algorithms, we have introduced different schemes and classifications. In particular, we have discussed diverse search strategies in structured, semi-structured, and un-

structured P2P networks. Finally, we have identified common features in these networks, carrying out a comparative analysis for contrasting these features.

As discussed in this article, semantic issues in intelligent query routing provide a significant added value for improving search in distributed environments. This survey aims at offering an in-depth analysis of the state of the art in this exciting research area, oriented towards a wide and heterogeneous audience of researchers and practitioners working on P2P networks. New recent advances in Artificial Intelligence techniques (e.g. [103]) show that future developments in P2P networks will allow to go beyond the traditional semantic analysis by adding qualitative reasoning capabilities to the nodes. Even though some motivating preliminary results have been obtained, most of the research work in this direction is still to be done.

Acknowledgments. We want to thank the reviewers who provided helpful suggestions and insights for improving the original version of the article. This research was supported by the projects PICT-ANPCyT 2014-0624, PIP-CONICET 112-2012010-0487, and PGI-UNS 24/N039.

References

1. Kazaa. http://www.kazaa.com. Retrieved in September 2011 (2011)
2. Buddycloud. http://buddycloud.com/. Retrieved in June 2017 (2017)
3. Diaspora. https://diasporafoundation.org/. Retrieved in June 2017 (2017)
4. Freenet. http://freenetproject.org. Retrieved in June 2017 (2017)
5. Friendica. http://friendi.ca/. Retrieved in June 2017 (2017)
6. Gnusocial. https://gnu.io/social/. Retrieved in June 2017 (2017)
7. Kune. http://kune.ourproject.org. Retrieved in June 2017 (2017)
8. Mastodon. https://mastodon.social. Retrieved in June 2017 (2017)
9. Minds. https://www.minds.com/. Retrieved in June 2017 (2017)
10. Twister. http://twister.net.co/. Retrieved in June 2017 (2017)
11. Apache cassandra. http://cassandra.apache.org. Retrieved in September 2018 (2018)
12. Bigtable. http://cloud.google.com/bigtable/. Retrieved in September 2018 (2018)
13. Dynamo. http://aws.amazon.com/es/dynamodb/. Retrieved in September 2018 (2018)
14. emule. www.emule-project.net. Retrieved in September 2018 (2018)
15. Facebook. http://facebook.com/. Retrieved in September 2018 (2018)
16. Myspace. http://myspace.com/. Retrieved in September 2018 (2018)
17. Popcorn time. https://popcorntime.sh/es. Retrieved in September 2018 (2018)
18. Twitter. http://twitter.com/. Retrieved in September 2018 (2018)
19. Windows azure storage. http://azure.microsoft.com. Retrieved in September 2018 (2018)
20. Aberer, K., Hauswirth, M.: An overview of peer-to-peer information systems. In: Workshop on Distributed Data and Structures. vol. 14, pp. 171–188 (2002)
21. Adamic, L.A., Lukose, R.M., Puniyani, A.R., Huberman, B.A.: Search in power-law networks. Physical Review E 64, 046135 (Sep 2001)
22. Akavipat, R., Wu, L.S., Menczer, F., Maguitman, A.G.: Emerging semantic communities in peer web search. In: Proceedings of the international workshop on Information retrieval in peer-to-peer networks. pp. 1–8. P2PIR '06, ACM, New York, NY, USA (2006)

23. Amad, M., Aïssani, D., Meddahi, A., Benkerrou, M., Amghar, F.: De bruijn graph based solution for lookup acceleration and optimization in p2p networks. Wireless Personal Communications 85(3), 1471–1486 (Dec 2015), https://doi.org/10.1007/s11277-015-2851-y
24. Androutsellis-Theotokis, S., Spinellis, D.: A survey of peer-to-peer content distribution technologies. ACM Computing Surveys (CSUR) 36(4), 335–371 (2004)
25. Androutsellis-Theotokis, S., Spinellis, D.: A survey of peer-to-peer content distribution technologies. ACM Comput. Surv. 36(4), 335–371 (Dec 2004), http://doi.acm.org/10.1145/1041680.1041681
26. Arour, K., Yeferny, T.: Learning model for efficient query routing in P2P information retrieval systems. Peer-to-Peer Networking and Applications 8(5), 741–757 (2015)
27. Azimi, R., Sajedi, H., Ghayekhloo, M.: A distributed data clustering algorithm in p2p networks. Applied Soft Computing 51, 147–167 (2017)
28. Babaei, H., Fathy, M., Romoozi, M.: Modeling and optimizing random walk content discovery protocol over mobile ad-hoc networks. Performance Evaluation 74, 18–29 (2014)
29. Baccelli, F., Mathieu, F., Norros, I., Varloot, R.: Can P2P networks be super-scalable? In: INFOCOM 2013. Annual Joint Conference of the IEEE Computer and Communications Societies. pp. 1753–1761. IEEE (2013)
30. Badis, L., Amad, M., Aîssani, D., Bedjguelal, K., Benkerrou, A.: Routil: P2p routing protocol based on interest links. In: Advanced Aspects of Software Engineering (ICAASE), 2016 International Conference on. pp. 1–5. IEEE (2016)
31. Bashmal, L., Almulifi, A., Kurdi, H.: Hybrid resource discovery algorithms for unstructured peer-to-peer networks. Procedia Computer Science 109, 289–296 (2017)
32. Bawa, M., Manku, G.S., Raghavan, P.: Sets: search enhanced by topic segmentation. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. pp. 306–313. ACM (2003)
33. Ben-Gal, I., Shavitt, Y., Weinsberg, E., Weinsberg, U.: Peer-to-peer information retrieval using shared-content clustering. Knowledge and information systems 39(2), 383–408 (2014)
34. Brienza, S., Cebeci, S.E., Masoumzadeh, S.S., Hlavacs, H., Özkasap, Ö., Anastasi, G.: A survey on energy efficiency in p2p systems: File distribution, content streaming, and epidemics. ACM Computing Surveys (CSUR) 48(3), 36 (2016)
35. Broder, A., Mitzenmacher, M.: Network applications of bloom filters: A survey. Internet mathematics 1(4), 485–509 (2004)
36. Castano, S., Montanelli, S.: Semantic self-formation of communities of peers. In: Workshop on Ontologies in Peer-to-Peer Communities. European Semantic Web Conference (2005)
37. Castro, M., Druschel, P., Ganesh, A., Rowstron, A., Wallach, D.S.: Secure routing for structured peer-to-peer overlay networks. ACM SIGOPS Operating Systems Review 36(SI), 299–314 (2002)
38. Chen, H., Gong, Z., Huang, Z.: Self-learning routing in unstructured P2P network. International Journal of Information Technology 11(12), 59–67 (2005)
39. Choi, J., Han, J., Cho, E., Kwon, T.T., Choi, Y.: A survey on content-oriented networking for efficient content delivery. Communications Magazine 49(3), 121–127 (2011)
40. Ciraci, S., Körpeoglu, I., Ulusoy, O.: Reducing query overhead through route learning in unstructured peer-to-peer network. Journal of Network and Computer Applications 32(3), 550–567 (May 2009)
41. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W.: Freenet: A distributed anonymous information storage and retrieval system. In: Designing Privacy Enhancing Technologies. pp. 46–66. Springer (2001)
42. Craswell, N., Hawking, D.: Web information retrieval, chap. 5, pp. 85–101. John Wiley & Sons, Ltd (2009)

43. Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. In: Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02). pp. 23–32. IEEE Computer Society (2002)
44. Crespo, A., Garcia-Molina, H.: Semantic overlay networks for P2P systems. In: Agents and Peer-to-Peer Computing, pp. 1–13. Springer (2005)
45. Daswani, N., Garcia-Molina, H., Yang, B.: Open problems in data-sharing peer-to-peer systems. In: Database Theory–ICDT 2003, pp. 1–15. Springer (2003)
46. de Bruijn, N.: A combinatorial problem. Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen. Series A 49(7), 758–764 (1946)
47. Douceur, J.R.: The sybil attack. In: International workshop on peer-to-peer systems. pp. 251–260. Springer (2002)
48. Du, A., Callan, J.: Probing a collection to discover its language model. Tech. rep., University of Massachusetts (1998)
49. Dunn, R.J., Zahorjan, J., Gribble, S.D., Levy, H.M.: Presence-based availability and P2P systems. In: Peer-to-Peer Computing, 2005. P2P 2005. Fifth IEEE International Conference on. pp. 209–216. IEEE (2005)
50. Einhorn, M.A., Rosenblatt, B.: Peer-to-peer networking and digital rights management: How market tools can solve copyright problems. J. Copyright Soc'y USA 52, 239 (2004)
51. Fanti, G., Viswanath, P.: Anonymity properties of the bitcoin p2p network. arXiv (2017), https://arxiv.org/abs/1703.08761
52. Felber, P., Kropf, P., Schiller, E., Serbu, S.: Survey on load balancing in peer-to-peer distributed hash tables. IEEE Communications Surveys and Tutorials 16(1), 473–492 (2014)
53. Fraigniaud, P., Gauron, P.: D2b: A de bruijn based content-addressable network. Theoretical Computer Science 355(1), 65 – 79 (2006), http://www.sciencedirect.com/science/article/pii/S0304397505009163, complex Networks
54. Galluccio, L., Morabito, G., Palazzo, S., Pellegrini, M., Renda, M.E., Santi, P.: Georoy: A location-aware enhancement to viceroy peer-to-peer algorithm. Computer Networks 51(8), 1998–2014 (2007), https://doi.org/10.1016/j.comnet.2006.09.017
55. Gkantsidis, C., Mihail, M., Saberi, A.: Random walks in peer-to-peer networks. In: INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies. vol. 1. IEEE (2004)
56. Gravano, L., Chang, K., Garcia-Molina, H., Paepcke, A.: Starts: Stanford protocol proposal for internet retrieval and search. Tech. rep., Stanford University, Stanford, CA, USA (1997)
57. Grumbach, S., Riemann, R.: Secure and trustable distributed aggregation based on kademlia. In: di Vimercati, S.D.C., Martinelli, F. (eds.) ICT Systems Security and Privacy Protection - 32nd IFIP TC 11 International Conference, SEC 2017, Rome, Italy, May 29-31, 2017, Proceedings. IFIP Advances in Information and Communication Technology, vol. 502, pp. 171–185. Springer (2017), https://doi.org/10.1007/978-3-319-58469-0_12
58. Haddi, F.L., Benchaïba, M.: A survey of incentive mechanisms in static and mobile P2P systems. Journal of Network and Computer Applications 58, 108–118 (2015)
59. Han, J., Haihong, E., Le, G., Du, J.: Survey on nosql database. In: Pervasive computing and applications (ICPCA), 2011 6th international conference on. pp. 363–366. IEEE (2011)
60. Hazel, S., Wiley, O.: Achord: A variant of the chord lookup service for use in censorship resistant peer-to-peer publishing systems (04 2002)
61. Heck, H., Kieselmann, O., Wacker, A.: Evaluating connection resilience for the overlay network kademlia. In: Lee, K., Liu, L. (eds.) 37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017, Atlanta, GA, USA, June 5-8, 2017. pp. 2581–2584. IEEE Computer Society (2017), https://doi.org/10.1109/ICDCS.2017.101
62. Herschel, S.: Indexing dynamic networks. In: Cremers, A.B., Manthey, R., Martini, P., Steinhage, V. (eds.) Lecture Notes in Informatics. vol. 65, pp. 429–433 (2005)
63. Hsu, C.Y., Wang, K., Shih, H.C.: Decentralized structured peer-to-peer network and load balancing methods thereof (May 2013), uS Patent 8,443,086

64. Huang, A.: Similarity measures for text document clustering. In: Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008). pp. 49–56 (2008)
65. Huang, X., Chen, L., Huang, L., Li, M.: Routing algorithm using skipnet and small-world for peer-to-peer system. In: Proceedings of the 4th International Conference on Grid and Cooperative Computing. pp. 984–989. GCC'05, Springer-Verlag, Berlin, Heidelberg (2005)
66. Ismail, H., Germanus, D., Suri, N.: P2p routing table poisoning: A quorum-based sanitizing approach. Computers & Security 65, 283–299 (2017)
67. Jawhar, I., Wu, J.: A two-level random walk search protocol for peer-to-peer networks. In: 8th World Multi-Conference on Systemics, Cybernetics and Informatics. pp. 1–5 (2004)
68. Jin, H., Ning, X., Chen, H., Yin, Z.: Efficient query routing for information retrieval in semantic overlays. In: 21st Annual ACM Symposium on Applied Computing (SAC'06). pp. 23–27. ACM Press (2006)
69. Joseph, S.: NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks. In: Gregori, E., Cherkasova, L., Cugola, G., Panzieri, F., Picco, G. (eds.) Web Engineering and Peer-to-Peer Computing, Lecture Notes in Computer Science, vol. 2376, pp. 202–214. Springer Berlin Heidelberg (2002)
70. Kaashoek, M.F., Karger, D.R.: Koorde: A simple degree-optimal distributed hash table. In: Kaashoek, M.F., Stoica, I. (eds.) Peer-to-Peer Systems II. pp. 98–107. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
71. Kalogeraki, V., Gunopulos, D., Zeinalipour-Yazti, D.: A local search mechanism for peer-to-peer networks. In: Proceedings of the Eleventh International Conference on Information and Knowledge Management. pp. 300–307. CIKM '02, ACM, New York, NY, USA (2002)
72. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in P2P networks. In: Proceedings of the 12th International Conference on World Wide Web. pp. 640–651. WWW '03, ACM, New York, NY, USA (2003)
73. Khambatti, M., Ryu, K.D., Dasgupta, P.: Structuring peer-to-peer networks using interest-based communities. In: International Workshop On Databases, Information Systems, and Peer-to-Peer Computing. pp. 48–63. Springer (2003)
74. Klampanos, I., Jose, J.: An evaluation of a cluster-based architecture for peer-to-peer information retrieval. Lecture Notes in Computer Science 4653, 380–391 (2007), http://eprints.gla.ac.uk/39573/
75. Kleinberg, J.: Complex networks and decentralized search algorithms. In: Proceedings of the International Congress of Mathematicians (ICM). vol. 3, pp. 1019–1044 (2006)
76. Kumar, A., Xu, J., Zegura, E.W.: Efficient and scalable query routing for unstructured peer-to-peer networks. In: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. vol. 2, pp. 1162–1173. IEEE (2005)
77. Kurose, J.F., Ross, K.: Computer Networking: A Top-Down Approach Featuring the Internet. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edn. (2002)
78. Lele, N., Wu, L.S., Akavipat, R., Menczer, F.: Sixearch.org 2.0 peer application for collaborative web search. In: Proceedings of the 20th ACM Conference on Hypertext and Hypermedia. pp. 333–334. HT '09, ACM, New York, NY, USA (2009)
79. Li, J., Khan, S.U., Ghani, N.: Semantics-Based Resource Discovery in Large-Scale Grids, pp. 409–430. John Wiley & Sons, Inc. (2013)
80. Li, R.H., Yu, J.X., Huang, X., Cheng, H.: Random-walk domination in large graphs. In: 30th International Conference on Data Engineering (ICDE). pp. 736–747. IEEE (2014)
81. Li, X., Wu, J.: Searching Techniques in Peer-to-Peer Networks, chap. 37, pp. 617–642. Auerbach Publications, Boston, MA, USA (2006)
82. Loach, S., Bowman, D.: Heuristics-based peer to peer message routing (May 20 2008), uS Patent 7,376,749
83. Löser, A., Staab, S., Tempich, C.: Semantic methods for P2P query routing. In: Multiagent System Technologies, pp. 15–26. Springer (2005)

84. Lu, J., Callan, J.: Full-text federated search of text-based digital libraries in peer-to-peer networks. Information Retrieval 9(4), 477–498 (2006)
85. Lu, J., Callan, J.: Content-based peer-to-peer network overlay for full-text federated search. In: Large Scale Semantic Access to Content (Text, Image, Video, and Sound). pp. 490–509. RIAO '07, Le Centre De Hautes Etudes Internationales D'informatique Documentaire, Paris, France (2007), http://dl.acm.org/citation.cfm?id=1931390.1931438
86. Lua, E.K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A survey and comparison of peer-to-peer overlay network schemes. Communications Surveys & Tutorials 7(2), 72–93 (Apr 2005)
87. Luo, B., Jin, Y., Luo, S., Sun, Z.: A symmetric lookup-based secure p2p routing algorithm. KSII Transactions on Internet & Information Systems 10(5), 2203–2217 (2016)
88. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer networks. In: Proceedings of the 16th international conference on Supercomputing. pp. 84–95. ACM (2002)
89. Malatras, A.: State-of-the-art survey on P2P overlay networks in pervasive computing environments. Journal of Network and Computer Applications 55, 1–23 (2015), http://www.sciencedirect.com/science/article/pii/S1084804515000879
90. Malkhi, D., Naor, M., Ratajczak, D.: Viceroy: a scalable and dynamic emulation of the butterfly. In: Ricciardi, A. (ed.) Proceedings of the Twenty-First Annual ACM Symposium on Principles of Distributed Computing, PODC 2002, Monterey, California, USA, July 21-24, 2002. pp. 183–192. ACM (2002), http://doi.acm.org/10.1145/571825.571857
91. Maymounkov, P., Mazières, D.: Kademlia: A peer-to-peer information system based on the XOR metric. In: Druschel, P., Kaashoek, M.F., Rowstron, A.I.T. (eds.) Peer-to-Peer Systems, First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2429, pp. 53–65. Springer (2002), https://doi.org/10.1007/3-540-45748-8_5
92. Menczer, F., Wu, L.S., Akavipat, R.: Intelligent peer networks for collaborative web search. AI Magazine 29(3), 35 (2008)
93. Meng, F., Ding, L., Peng, S., Yue, G.: A P2P network model based on hierarchical interest clustering algorithm. Journal of Software 8(5), 1262–1267 (May 2013)
94. Meng, X.: speedtrust: a super peer-guaranteed trust model in hybrid p2p networks. The Journal of Supercomputing pp. 1–28 (2018)
95. Michlmayr, E., Graf, S., Siberski, W., Nejdl, W.: Query routing with ants. In: Workshop on Ontologies in Peer-to-Peer Communities. European Semantic Web Conference (2005)
96. Morr, D.: Lionshare: A federated p2p app. In: Internet2 members meeting (2007)
97. Nah, F.F.H.: A study on tolerable waiting time: how long are web users willing to wait? Behaviour & Information Technology 23(3), 153–163 (2004)
98. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008)
99. Naor, M., Wieder, U.: Novel architectures for p2p applications: the continuous-discrete approach. ACM Transactions on Algorithms (TALG) 3(3), 34 (2007)
100. Napster: http://free.napster.com (2011)
101. Nascimento, M.A.: Peer-to-peer: Harnessing the power of disruptive technologies. ACM SIGMOD Record 32(2), 57–58 (Jun 2003)
102. Nicolini, A.L., Lorenzetti, C.M., Maguitman, A.G., Chesñevar, C.I.: Intelligent algorithms for reducing query propagation in thematic P2P search. In: Anales del XIX Congreso Argentino de Ciencias de la Computación (CACIC). pp. 71–79. Mar del Plata, Buenos Aires, Argentina (Oct 2013)
103. Nicolini, A.L., Maguitman, A.G., Chesñevar, C.I.: Argp2p: An argumentative approach for intelligent query routing in P2P networks. In: Theory and Applications of Formal Argumentation - Third International Workshop, TAFA 2015, Buenos Aires, Argentina, July 25-26, 2015, Revised Selected Papers. pp. 194–210 (2015)

104. Okubo, T., Ueda, K.: Peer-to-peer contents delivery system considering network distance. In: Network Operations and Management Symposium (APNOMS), 2011 13th Asia-Pacific. pp. 1–4. IEEE (2011)
105. Passarella, A.: A survey on content-centric technologies for the current internet: CDn and P2P solutions. Computer Communications 35(1), 1–32 (2012)
106. Poenaru, A., Istrate, R., Pop, F.: Aft: Adaptive and fault tolerant peer-to-peer overlaya user-centric solution for data sharing. Future Generation Computer Systems 80, 583–595 (2018)
107. Qamar, M., Malik, M., Batool, S., Mehmood, S., Malik, A.W., Rahman, A.: Centralized to Decentralized Social Networks: Factors that Matter, chap. 3, pp. 37–54. IGI Global (2016)
108. Qin, C., Yang, Z., Liu, H.: User interest modeling for P2P document sharing systems based on k-medoids clustering algorithm. In: Seventh International Joint Conference on Computational Sciences and Optimization (CSO). pp. 576–578. IEEE (2014)
109. Rabin, M.O.: Efficient dispersal of information for security, load balancing, and fault tolerance. J. ACM 36(2), 335–348 (Apr 1989), http://doi.acm.org/10.1145/62044.62050
110. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A scalable content-addressable network. SIGCOMM Computer Communication Review 31(4), 161–172 (Aug 2001)
111. Ripeanu, M.: Peer-to-peer architecture case study: Gnutella network. In: Proceedings of the First International Conference on Peer-to-Peer Computing. pp. 99–100 (2001)
112. Risson, J., Moors, T.: Survey of research towards robust peer-to-peer networks: search methods. Computer Networks 50(17), 3485–3521 (2006)
113. Rosenfeld, A., Goldman, C.V., Kaminka, G.A., Kraus, S.: Phirst: A distributed architecture for P2P information retrieval. Information Systems 34(2), 290–303 (2009)
114. Rowstron, A., Druschel, P.: Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In: Middleware 2001. pp. 329–350. Springer (2001)
115. Schlosser, M., Sintek, M., Decker, S., Nejdl, W.: A scalable and ontology-based P2P infrastructure for semantic web services. In: Second International Conference on Peer-to-Peer Computing. pp. 104–111. IEEE (2002)
116. Schmidt, C., Parashar, M.: A peer-to-peer approach to web service discovery. World Wide Web 7(2), 211–229 (2004)
117. Schollmeier, R.: A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In: Proceedings of the First International Conference on Peer-to-Peer Computing. pp. 101–102. P2P '01, IEEE Computer Society, Washington, DC, USA (2001)
118. Shah, V., de Veciana, G., Kesidis, G.: A stable approach for routing queries in unstructured p2p networks. IEEE/ACM Transactions on Networking 24(5), 3136–3147 (2016)
119. Sharan, A.: Exploiting semantic locality to improve peer-to-peer search mechanisms. Ph.D. thesis, Rochester Institute of Technology (2006)
120. Shen, X.J., Chang, Q., Gou, J.P., Mao, Q.R., Zha, Z.J., Lu, K.: Collaborative q-learning based routing control in unstructured P2P networks. In: MultiMedia Modeling. pp. 910–921. Springer (2016)
121. Shokouhi, M., Zobel, J., Tahaghoghi, S., Scholer, F.: Using query logs to establish vocabularies in distributed information retrieval. Information Processing and Management 43(1), 169180 (2007), http://research.microsoft.com/apps/pubs/default.aspx?id=80270
122. da Silva, P.M., Dias, J., Ricardo, M.: Mistrustful p2p: Deterministic privacy-preserving p2p file sharing model to hide user content interests in untrusted peer-to-peer networks. Computer Networks 120, 87–104 (2017)
123. Sripanidkulchai, K., Maggs, B., Zhang, H.: Efficient content location using interest-based locality in peer-to-peer systems. In: Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. vol. 3, pp. 2166–2176. IEEE (Mar 2003)

124. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOMM Computer Communication Review 31(4), 149–160 (2001)
125. Suel, T., Mathur, C., wen Wu, J., Zhang, J., Delis, A., Kharrazi, M., Long, X., Shanmugasundaram, K.: Odissea: A peer-to-peer architecture for scalable web search and information retrieval. In: International Workshop on the Web and Databases. pp. 67–72 (2003)
126. Sugiura, A., Etzioni, O.: Query routing for web search engines: Architecture and experiments. Computer Networks 33(1), 417–429 (2000)
127. Tang, C., Xu, Z., Mahalingam, M.: psearch: Information retrieval in structured overlays. ACM SIGCOMM Computer Communication Review 33(1), 89–94 (Jan 2003)
128. Tempich, C., Staab, S., Wranik, A.: Remindin': Semantic query routing in peer-to-peer networks based on social metaphors. In: Proceedings of the 13th International Conference on World Wide Web. pp. 640–649. WWW '04, ACM, New York, NY, USA (2004)
129. Tigelaar, A.S., Hiemstra, D., Trieschnigg, D.: Peer-to-peer information retrieval: An overview. ACM Transactions on Information Systems 30(2), 9:1–9:34 (May 2012)
130. Tirado, J.M., Higuero, D., Isaila, F., Carretero, J., Iamnitchi, A.: Affinity P2P: A self-organizing content-based locality-aware collaborative peer-to-peer network. Computer Networks 54(12), 2056–2070 (2010)
131. Tsoumakos, D., Roussopoulos, N.: Adaptive probabilistic search for peer-to-peer networks. In: Third International Conference on Peer-to-Peer Computing. pp. 102–109. IEEE (2003)
132. Tsoumakos, D., Roussopoulos, N.: Analysis and comparison of p2p search methods. In: Proceedings of the 1st international conference on Scalable information systems. p. 25. ACM (2006)
133. Ueda, K., Akase, J.i., Okubo, T.: Analysis of peer cluster layers selection criteria for P2P contents distribution systems. In: 15th Asia-Pacific Network Operations and Management Symposium (APNOMS). pp. 1–6 (2013)
134. Ueda, K., Okubo, T.: Peer-to-peer contents distribution system using multiple peer clusters. In: 14th Asia-Pacific Network Operations and Management Symposium (APNOMS). pp. 1–6 (2012)
135. Voulgaris, S., Kermarrec, A., Massoulié, L., van Oteen, M.: Exploiting semantic proximity in peer-to-peer content searching. In: Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems. pp. 238–243. IEEE Computer Society, Washington, DC, USA (2004)
136. Wallach, D.S.: A survey of peer-to-peer security issues. In: Okada, M., Pierce, B.C., Scedrov, A., Tokuda, H., Yonezawa, A. (eds.) Software Security — Theories and Systems. pp. 42–57. Springer Berlin Heidelberg, Berlin, Heidelberg (2003)
137. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393(6684), 440–442 (1998)
138. Wu, K., Wu, C.: State-based search strategy in unstructured P2P. Future Generation Computer Systems 29(1), 381–386 (2013)
139. Wu, L.S., Akavipat, R., Menczer, F.: 6S: Distributing crawling and searching across web peers. In: Web Technologies, Applications, and Services. pp. 159–164 (2005)
140. Yan, F., Zhan, S.: A peer-to-peer approach with semantic locality to service discovery. In: Jin, H., Pan, Y., Xiao, N., Sun, J. (eds.) Grid and Cooperative Computing - GCC 2004, Lecture Notes in Computer Science, vol. 3251, pp. 831–834. Springer Berlin Heidelberg (2004)
141. Yang, B., Garcia-Molina, H.: Improving search in peer-to-peer networks. In: Proceedings 22nd International Conference on Distributed Computing Systems. pp. 5–14 (July 2002)
142. Yang, B., Garcia-Molina, H.: Improving search in peer-to-peer networks. In: 22nd International Conference on Distributed Computing Systems. pp. 5–14. IEEE (2002)
143. Yang, Y., Dunlap, R., Rexroad, M., Cooper, B.F.: Performance of full text search in structured and unstructured peer-to-peer systems. In: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. IEEE Press (2006)

144. Yang, Z., Xing, Y., Chen, C., Xue, J., Dai, Y.: Understanding the performance of offline download in real P2P networks. Peer-to-Peer Networking and Applications 8(6), 992–1007 (2015)
145. Yu, W., Lin, X.: IRwr: incremental random walk with restart. In: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. pp. 1017–1020. ACM (2013)
146. Yu, Y.T., Gerla, M., Sanadidi, M.: Scalable vanet content routing using hierarchical bloom filters. Wireless Communications and Mobile Computing 15(6), 1001–1014 (2015)
147. Zeinalipour-Yazti, D., Kalogeraki, V., Gunopulos, D.: Information retrieval techniques for peer-to-peer networks. Computing in Science Engineering 6(4), 20–26 (2004)
148. Zeng, B., Wang, R.: A novel lookup and routing protocol based on can for structured p2p network. In: Computer Communication and the Internet (ICCCI), 2016 IEEE International Conference on. pp. 6–9. IEEE (2016)
149. Zhao, B.Y., Huang, L., Stribling, J., Rhea, S.C., Joseph, A.D., Kubiatowicz, J.D.: Tapestry: A resilient global-scale overlay for service deployment. Journal on Selected Areas in Communications 22(1), 41–53 (2004)
150. Zhu, Y., Wang, H., Hu, Y.: Integrating semantics-based access mechanisms with P2P file systems. In: Third International Conference on Peer-to-Peer Computing. pp. 118–125 (Sep 2003)
151. Zhu, Y., Hu, R., Fei, L.: A low latency resource location algorithm for unstructured P2P networks. In: International Conference on Computational Intelligence and Software Engineering. pp. 1–4. IEEE (2010)
152. Zhuge, H., Liu, J., Feng, L., Sun, X., He, C.: Query routing in a peer-to-peer semantic link network. Computational Intelligence 21(2), 197–216 (2005)

Ana L. Nicolini is a Teaching Assistant and a Postdoctoral fellow at the Computer Science and Engineering Department at Universidad Nacional del Sur in Argentina. In 2012 he obtained a Computer Science degree at Universidad Nacional del Sur. In April 2013 he started his doctoral thesis research with a fellowship granted by CONICET, and obtained his PhD degree in Computer Science in December 2017. Her research interests include information retrieval, argumentation and complex networks.

Carlos M. Lorenzetti is an Adjunt Researcher at the National Council for Science and Technology (CONICET) of Argentina and a Professor at the Department of Computer Science and Engineering of the Universidad Nacional del Sur (Argentina). He obtained his PhD in Computer Science at Universidad Nacional del Sur in 2011. His research interests include datamining, knowledge discovery and information retrieval.

Ana G. Maguitman is an Independent Researcher at the National Council for Science and Technology (CONICET) of Argentina and a Professor at the Department of Computer Science and Engineering of the Universidad Nacional del Sur (Argentina). She obtained her PhD in Computer Science at Indiana University (USA). Dr. Maguitman leads the Knowledge Management and Information Retrieval Research Group at Universidad Nacional del Sur (http://ir.cs.uns.edu.ar/). Her research is focused on intelligent information retrieval and text mining.

Carlos I. Chesñevar is a Principal Researcher from the National Council for Science and Technology (CONICET), Argentina, and the Director of the Research Institute for

Computer Science and Engineering (ICIC) of the Universidad Nacional del Sur in Bahía Blanca, Argentina. He has led and participated in several scientific projects related to artificial intelligence and e-government supported by different funding agencies (DAAD Germany, CONICET Argentina, Microsoft Research Latinamerica, etc.).

Received: April 11, 2018; Accepted: January 10, 2019.

Computer Science and Information Systems 16(2):443–468 https://doi.org/10.2298/CSIS180428003L

Dimension Reduction and Classification of Hyperspectral Images based on Neural Network Sensitivity Analysis and Multi-instance Learning

Hui Liu[1,2], Chenming Li[1] and Lizhong Xu[1]

1 College of Computer and Information Engineering, Hohai University, Nanjing, 211100, China, 9120030059@jxust.edu.cn
2 School of Science, Jiangxi University of Science and Technology, Ganzhou, 341000, China, lcm@hhu.edu.cn
1 College of Computer and Information Engineering, Hohai University, Nanjing, 211100, China, lzhxu@hhu.edu.cn

Abstract. Hyperspectral remote image sensing is a rapidly developing integrated technology used widely in numerous areas. The rich spectral information from hyperspectral images aids in recognition and classification of many types of objects, but the high dimensionality of these images leads to information redundancy. In this paper, we used sensitivity analysis for dimension reduction. However, another challenge is that hyperspectral images identify objects as either a "different body with the same spectrum" or "same body with a different spectrum." Therefore, it is difficult to maintain the correct correspondence between ground objects and samples, which hinders classification of the images. This issue can be addressed using multi-instance learning for classification. In our proposed method, we combined neural network sensitivity analysis with a multi-instance learning algorithm based on a support vector machine to achieve dimension reduction and accurate classification for hyperspectral images. Experimental results demonstrated that our method provided strong overall classification effectiveness when compared with prior methods.

Keywords: Sensitivity Analysis, Artificial Neural Network, Ruck Sensitivity Analysis, Dimension Reduction, Classification, Hyperspectral Images, Multi-instance Learning, SVM

1. Introduction

Hyperspectral remote sensing technology is a rapidly developing field used in numerous areas of specialty, ranging from astronomy and geology to medicine and aerial surveillance, among others. Hyperspectral images provide rich spectral information that can aid in the recognition and classification of ground objects. However, the high dimensionality of these images leads to costly information redundancy, so the dimensions must be reduced. Sensitivity analysis using neural networks can be used for dimension reduction. However, another challenge remains. Hyperspectral images identify objects as either a "different body with the same spectrum" or "same body with a different spectrum." Therefore, it is difficult to maintain the correct correspondence

between ground objects and samples, which makes classification of the images difficult and computationally costly. This issue can be addressed using multi-instance learning for classification. In this research, we combine neural network sensitivity analysis with a multi-instance learning algorithm based on a support vector machine (SVM) to achieve dimension reduction for hyperspectral remote sensing images.

Hyperspectral imaging differs from general multispectral imaging insofar as hyperspectral imaging can display two-dimensional spatial information for a region of interest (e.g., the earth's surface), and it can add a dimension of spectral information. Multispectral images include only a few spectral bands, whereas hyperspectral images include hundreds of bands that are much narrower. Therefore, hyperspectral images can form an "image cube" [1]. In addition to multiple bands, hyperspectral images have distinct characteristics such as large amounts of data and high information redundancy that present difficulties for storing, transmitting, and processing the images. As a result, a band selection operation must be performed to reduce some of the unnecessary bands before processing hyperspectral images [2]. This operation helps to decrease the calculation requirements for hyperspectral image classification, and effectively avoids the Hughes phenomenon [3].

Hyperspectral image band selection can be considered an NP-hard combinatorial optimization problem [4]. Typically, search algorithms are used to search a band subset, allowing the evaluation standard to achieve its optimal value across all of the bands of the hyperspectral image. This queried band subset can then be treated as the optimal band combination. The common method of band selection and dimension reduction [5] for hyperspectral remote sensing images is to select several bands from the whole band to represent the whole band space. This approach requires that the band combination selected be able to provide effective improvement for classification accuracy in the subsequent classification. For this research, we select the band according to the band’s contribution to classification. Bands that help to improve classification accuracy are selected first.

For classification of hyperspectral remote sensing images, neural network classifiers are commonly used because they work well for classifying images with high dimensionality and nonlinear structures. To provide a quantitative evaluation of the effects of one band on classification accuracy, a neural network sensitivity analysis can be employed that is based on a neural network classifier. Sensitivity analysis by a neural network [6] can provide a quantitative description of the influence of the input variables of a model on the output variables. The sensitivity coefficient of the model properties is sorted. Properties with larger sensitivity coefficients are chosen, and those with smaller ones are no longer included. In this way, the model is simplified, and the complexity of model processing is reduced. In this paper, we apply neural network sensitivity analysis to the band selection for hyperspectral remote sensing images, combined with a frequently used BP neural network classifier.

In addition to dimension reduction, the issue remains that hyperspectral images identify objects as either a "different body with the same spectrum" or "same body with different spectrum," which means they are unable to maintain the correct correspondence between target objects and samples. This challenge can be addressed using multi-instance learning for classification of hyperspectral images. In the mid- and late 1990s, T. G. Dietterich et al. proposed the concept of multi-instance learning in the study of drug activity prediction [7]. In this kind of learning, the training set consists of

several “bags” labeled with concept tags. Each bag contains several instances without concept notations. A bag is labeled positive if it contains at least one instance that is positive, and a bag is labeled negative if all of the instances in it are negative. By learning from the training bags, the learning system can predict the concept tag of a bag that is outside the training set as correctly as possible. Following on the work of T. G. Dietterich et al., many researchers began to devise practical multi-instance learning algorithms. Multi-instance learning stirred great interest in the machine learning field because it was a promising new learning framework in an area of machine learning previously unexplored. Multi-instance learning has unique properties and continues to offer good prospects for wide application.

A pixel-level classifier can help to divide the remotely sensed hyperspectral imagery, but if spectral background noise and clutter noise are present, classification accuracy will be decreased because of omissions or faults. Recently, some researchers proposed to adopt classification based on pattern spot images to solve the omissions or faults brought by spectrum changes. A pattern spot image refers to the single zone whose shape shares the same features with a spectrum. CHEN Jie et al. [8] put forward the rough set theory-based object-oriented classification of high resolution remotely sensed imagery. First, they abstracted the pattern spot image by means of watershed segmentation. Next, they analyzed the texture features of the abstracted images using Gabor wavelets, and divided the texture classification rules for abstracted images based on rough set theory. ZHANG Chuan et al. put forward object-oriented classification of high resolution remotely sensed imagery [9]. YANG Chang-bao et al. explored the object-oriented classification of remotely sensed imagery, and they determined the classification by segmenting the orthorectification SPOT image based on a distributed domain solver [10]. TAN Yu-min et al. put forward an object-oriented remote sensing image segmentation approach based on edge detection. [11]. Pattern spot image classification differs greatly from pixel-based classification because the former includes various regional texture space information, while the latter has only spectral features. Pixel-based classification considers only the features of a single pixel. Classification based on pattern spot images is less likely to be disturbed by spectral background noise and is more likely to retain regional integrity since it is abstracted by the regional spectra and space characteristics.

However, pattern spot image classification still has weaknesses in terms of its anti-noise properties that result in low classification accuracy with accompanying omissions or faults. Since the anti-noise property fails to provide the needed filtering, the classification will be less likely to be disturbed by noise with simulti-instance learningar spectra and space characteristics classified in the same zone, which is bigger than the object by making use of the regional relevance and object-oriented basis. DU Pei-jun et al. proposed that the cases in which different objects may have the same spectra characteristics or the same object may have different spectra characteristics, together with the noise in training instances, can be regarded as particular representatives of “ambiguity” in the training bag multi-instance learning. Therefore, when objects from the image segmentation are regarded as an instance in multi-instance learning, the object set of clutter is regarded as the bag, and multi-instance learning can be used in remote sensing classification [12].

The phenomena by which the same objects may have different spectra and different objects may have the same spectrum are the main source of land surface complexity for

remote sensing images. The complexity of land surface composition and the difficulty in selecting training samples cause the classification process to be highly dependent on human experience and prior knowledge. When using sensitivity analysis provided by an artificial neural network to realize dimension reduction for hyperspectral images, all of the bands are divided into several groups, as long as a lower correlation exists between adjacent bands. In addition, a differential evolution (DE) algorithm is used for optimizing the neural network structure. The bands that make only small contributions are given up based on the Ruck sensitivity analysis method.

Given the special advantages of multi-instance learning for solving ambiguous problems, and the advantages of neural network sensitivity analysis for dimension reduction, we suggest that integrating both multi-instance learning and sensitivity analysis for hyperspectral image classification can reduce the uncertainty of classification results. In view of this background and the benefits of applying new machine learning methods to remote sensing image classification, in this paper, we combine multiple-instance learning and an ensemble artificial neural network with embedded sensitivity analysis to improve the accuracy of hyperspectral image classification.

2. Related Work

2.1. Neural Network Sensitivity Analysis

Sensitivity analysis is an important research focus in the field of neural networks. In some practical applications, the availability of a huge amount of data can cause the trained neural network to become increasingly complex. The main task of sensitivity analysis is to determine how to analyze the parameters of the neural network effectively and simplify the scale of the network. Toward this end, we use the following procedure in this research.

Assume that the model is $y = f(x_1, x_2, ..., x_n)$, where x_i is the kth property value of the model. It is necessary to ensure that any changes of each property are within the possible value range. The next step is to study and predict the influence of the change(s) of these properties on the output value of the model [13]. The degree of influence is called the sensitivity coefficient of property x_i on output value y: the higher the sensitivity coefficient, the greater the influence of the property on the output value. Thus, the sensitivity analysis can provide a quantitative description of the influence of the input variables on the output variable of a model. Furthermore, we can sort the sensitivity coefficients of the model's properties. We can choose the properties with larger sensitivities and give up the smaller ones according to the practical considerations of the problems. In this way, the model can be simplified and the computational complexity of processing the model reduced, which means dimension reduction is achieved.

Sensitivity analysis works based on the specific model. In most cases, however, when people face vast amounts of information, they are not clear about how the internal

mechanisms of the data work. In such cases, they cannot build the model expression $y = f(x)$ directly, and therefore cannot conduct a sensitivity analysis either. However, researchers have shown that while a neural network does not need to model the physical concept of the research question, the neural network can provide more effective solutions for problems involving uncertainty or nonlinearity. The network provides a black box analysis model, and outputs reasonable results through the training and learning of input samples [14]. If the input data and output data are known, the neural network will use many simple neurons to simulate the non-linear relationships between the data. Neural network sensitivity analysis also uses the connection weights and the threshold between neurons to assess the influence of the input data on the output data [15].

Neural network sensitivity analysis can be divided into local sensitivity analysis and global sensitivity analysis. Some scholars have focused mainly on the study of local sensitivity analysis. There are four types of classical neural network sensitivity analysis. The first type is the sensitivity analysis method based on the connection weight, such as the Garson algorithm put forward in the early 1990s by Garson [16] and the Tchaban algorithm proposed by Tchaban [17]. The second type is sensitivity analysis based on the influence of the partial derivatives of output variables on input variables, such as Dimoponlos sensitivity analysis [18] and Ruck sensitivity analysis [19]. The third type is sensitivity analysis combined with statistics, such as methods based on random testing by Olden et al.[20]. The forth type is sensitivity analysis based on input variable disturbance, such as the method of adding white noise to the input data of a network and calculating the resulting change of output variables, an example of which was put forward by Scardi [21]

2.2. SVM Classification Methods and Multi-instance Learning

Training data are required to train the SVM model. However, these data cannot be separated without errors. The data points that are closest to the hyperplane are used to measure the margin, while the SVM attempts to identify the hyperplane that maximizes the margin and minimizes a quantitative proportion to the number of misclassification errors [22]. The SVM derives the optimal hyperplane as the solution of the following convex quadratic programming problem [23]:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C\sum_{i=1}^{n} \xi_i \quad s.t.\ y_i\left(w^T x_i + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0, i = 1,2,\cdots,n \quad , \quad (1)$$

where $\{(x_1, y_1),\ldots,(x_i, y_i)\}$ are the labeled training datasets with $x_i \in R^d$ and $y_i \in \{-1,1\}$; w^* and b^* define a linear classifier in the feature space; C is the regularization parameter defined by the user; and ξ_i is a positive slack variable that handles permitted errors.

The optimization problem can be reformulated through a Lagrange function, where Lagrange multipliers can be found via dual optimization to generate a convex quadratic programming solution as follows [24] [25]:

$$\max Q(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j x_i x_j K(x_i, x_j),\ 0 \le \alpha_i \le C, i = 1, 2 \cdots n \quad s.t. \sum_{i=1}^{n} \alpha_i x_i = 0 \quad , \quad (2)$$

where $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_n]$ is the vector of the Lagrange multipliers, while $K(\square, \square)$ is a kernel function. For a linearly non-separable case, a kernel function is introduced that satisfies the condition stated by Mercer's theorem and that corresponds to some types of inner product in the transformed (higher) dimensional feature space, as shown:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \quad . \quad (3)$$

The final result is a discrimination function $F(x)$ conveniently expressed as a function of the data in the original (lower) dimensional feature space:

$$F(x) = \text{sgn}[(w^*)^T \phi(x) + b^*] = \text{sgn}(\sum_{i=1}^{n} \alpha_i^* y_i K(x_i, x) + b^*) \quad . \quad (4)$$

Some popular kernel functions include the following:

a) Linear kernel

$$K(x, x_i) = (x \cdot x_i) \quad , \quad (5)$$

Polynomial kernel

$$K(x, x_i) = [(x^T x_i) + 1]^q \quad , \quad (6)$$

where q is a constant.

b) Gaussian Radial Basis Function kernel

$$K(x, x_i) = \exp\left\{ -\frac{\|x - x_i\|^2}{\gamma^2} \right\}, \gamma > 0 \quad , \quad (7)$$

Sigmoid kernel

$$k(x, x_i) = \tanh[v(x^T x_i) + c], \quad v > 0, c < 0 \quad . \quad (8)$$

Multi-instance learning was proposed as a new machine learning method to solve problems with relationships such as "1:N:1" using "object:description:label." In multi-instance learning, labeled bags composed of several unlabeled instances are treated as training samples, and the goal of learning is to predict the labels of unknown new bags. In contrast to traditional supervised learning, the bag strategy employed by multi-instance learning offers special advantages in dealing with ambiguous problems. Multi-instance learning is viewed as the fourth machine learning framework, in parallel with reinforcement learning, supervised learning, and unsupervised learning (Bolton & Gader, 2011; Lozano-Perez, 1998; Bolton, et al., 2011; Zhang, et al., 2004; Zhou, et al., 2002; Andrews, et al., 2002; Zhang, et al., 2010). At the moment research on multi-instance learning has focused mainly on creating new multi-instance learning algorithms and on designing applications of multi-instance learning for various fields (Zhang, et al., 2004; Zhou, et al., 2002; Li, et al., 2004). In the image processing field, multi-instance learning has been used for image retrieval and scene classification, with good results obtained in some existing experiments (Li, et al., 2007; Li, et al., 2010; Wang, et al., 2010; Li, et al., 2008) [26].

Traditional supervised learning can be treated as a special case of multi-instance learning. The transformation of traditional supervised learning algorithms to make them capable of dealing with multiple instance problems is an important branch in multi-instance learning algorithm research. Considering the bag concept, multi-instance learning can be viewed as a generalization of traditional supervised learning. Combining multiple learners for the purpose of enhancing the performance of the base learner is an effective method in traditional supervised learning frameworks. According to the supervised nature and classification function of multi-instance learning, multiple instance ensemble learning is also a feasible approach, and some researchers have shown that performance is increased by using an ensemble. Research into the integration of ensemble learning and multi-instance learning is an active branch of machine learning, so advances have been seen in remote sensing image classification (Qi, et al., 2011; Zhou, et al., 2003; Auer & Ortner, 2004; Kittler, et al., 1998).

Multi-instance learning algorithms based on an SVM can be divided into two categories: multi-instance learning based on samples (mi-SVM), and multi-instance learning based on bags (MI-SVM) [27–28]. mi-SVM tries to identify a maximal margin hyperplane for the instances, subject to the constraint that at least one instance of each positive bag is located in the positive half-space while all instances of negative bags are in the negative half-space. MI-SVM tries to identify a maximal margin hyperplane for the bags by regarding the margin of the "most positive instance" in a bag as the margin of that bag.

$$mi-SVM: \min_{\{y_i\}} \min \frac{1}{2}\|w\|^2 + C\sum_i \xi_i, \quad s.t. \ \ \forall i: y_i\left(\langle w, x_i\rangle + b\right) \geq 1-\xi_i, \xi_i \geq 0, y_i \in \{-1,1\}$$

$$\sum_{i\in I} \frac{y_i+1}{2} \geq 1, \forall I \ \ s.t. \ Y_I = 1, and \ \ y_i = -1, \quad \forall I \ \ s.t. Y_I = -1 \quad . \ (9)$$

$$MI-SVM: \min_{\{w,b,\xi\}} \frac{1}{2}\|w\|^2 + C\sum_I \xi_i, \ \ s.t. \ \ \forall I: Y_i \max_{\{i\in I\}}\left(\langle w, x_i\rangle + b\right) \geq 1-\xi_I, \xi_I \geq 0 \quad , \qquad (10)$$

where w and b are two parameters; ξ_i is a positive slack variable; x_i is the input value; $y_i \in \{-1,1\}$ is the output value; and C is the regularization parameter defined by the user.

3. Application of Neural Network Sensitivity Analysis and Multi-instance Learning to Band Selection

The back propagation (BP) neural network classifier is a common tool for the classification of hyperspectral remote sensing images. Combined with the neural network sensitivity analysis method introduced above, the band selection for BP can be carried out in the whole band space. The band combination that provides a large contribution to the classification is selected (as explained in Section 3.1 below), thereby realizing the purpose of dimension reduction. To get better analysis results, before conducting a sensitivity analysis the hyperspectral remote sensing image data should be pretreated. Some band combinations with weak correlation are preselected by subspace-partition. To avoid blindness in the selection of the initial weights and threshold of BP neural network [29], a differential evolution algorithm is used to optimize the BP neural

network. Last, the optimized BP neural network is used to conduct the sensitivity analysis. The sensitivity analysis results for all of the test samples are combined by using the comprehensive evaluation function, and finally the band with the biggest influence on classification results is selected. The specific process is explained in the following subsections of this paper.

3.1. Data Preprocessing and Band Selection

For the proposed method, data processing takes place using the following steps. Before employing the neural network sensitivity analysis, the original hyperspectral remote sensing image needs to be preprocessed to eliminate the bands that have interference from noise, water vapor, or other serious pollution. Select the object that has the largest number of samples as the pre-selected object that is good for classification. Typically, the original hyperspectral remote sensing image has a large number of bands, but there is a high correlation and high redundancy between bands. Therefore, to attain a good result from the sensitivity analysis, it is very important to choose as the input the band that also has a weak correlation.

To solve the above problem, the approach is to divide the whole band into several subspaces, and then select the band. The adaptive subspace decomposition (ASD) [30] method based on correlation filtering is used to divide the band set of the hyperspectral remote sensing image. First, we calculate the correlation coefficient denoted as R_{ij} between the two bands. Let u_i and u_j denote the number of *i* and *j* bands, respectively. As the value of the correlation coefficient grows larger, the correlation between bands becomes stronger, and as the correlation comes closer to 0, the correlation becomes weaker. R_{ij} is defined as:

$$R_{i,j} = \frac{E\left\{\left(x_i - \mu_i\right)\left(x_i - \mu_i\right)\right\}}{\sqrt{E\left\{\left(x_i - \mu_i\right)^2\right\}}\sqrt{E\left\{\left(x_j - \mu_j\right)^2\right\}}}. \quad (11)$$

The value R_{ij} of the matrix R ranges between 0 and 1. As Rij comes closer to 1, the correlation between the two bands becomes stronger. μ_i and μ_j are the mean values of x_i and x_j, respectively. They denote the gray mean values of the two bands. $E[\bullet]$ is the mathematical value expected. When all R_{ij} values are identified, then the proper threshold T_b is set. The continuous bands of $|R_{ij}| \geq T_b$ form a new subspace. We can control the number of subspaces and the number of bands in each subspace dynamically by changing the threshold T_b. Furthermore, *Rs* denotes the ratio of the number of bands in a subspace to the total number of bands in all subspaces. We select the bands in each subspace to create band combinations according to the ratio *Rs*. Then we reduce the correlation of the bands as much as possible. Combining the bands with the pre-object types and the object information of the original remote sensing image, we can determine the training sample P for input, expected value T for output, test sample P_Test for

input, and expected value T_Test for output, all of which are required for the training of a BP neural network. These steps make it convenient to determine the topological structure of the neural network.

3.2. Dimension Reduction and Classification based on Sensitivity Analysis and Multi-instance Learning

3.2.1 Ruck sensitivity analysis

Ruck sensitivity analysis (1990) is based on the partial derivatives of output variables for input variables. This method is designed for feedback neural networks, such as BP neural networks and RBF neural networks. This approach, which is simple and fast, evaluates the partial derivatives with the activation function of the neural network, and calculates the influence of the input data on the output data. Therefore, Ruck sensitivity analysis is used to study band selection for a hyperspectral remote sensing image based on a BP neural network classifier.

The value R_{ij} of the matrix R ranges between 0 and 1. As Rij comes closer to 1, the correlation between the two bands becomes stronger. μ_i and μ_j are the mean values of x_i and x_j, respectively. They denote the gray mean values of the two bands. $E[\bullet]$ is the mathematical value expected. When all R_{ij} values are identified, then the proper threshold T_b is set. The continuous bands of $|R_{ij}| \geq T_b$ form a new subspace. We can control the number of subspaces and the number of bands in each subspace dynamically by changing the threshold T_b. Furthermore, *Rs* denotes the ratio of the number of bands in a subspace to the total number of bands in all subspaces. We select the bands in each subspace to create band combinations according to the ratio *Rs*. Then we reduce the correlation of the bands as much as possible. Combining the bands with the pre-object types and the object information of the original remote sensing image, we can determine the training sample P for input, expected value T for output, test sample P_Test for input, and expected value T_Test for output, all of which are required for the training of a BP neural network. These steps make it convenient to determine the topological structure of the neural network.

The above Ruck sensitivity analysis is used only for the sensitivity analysis of the input value to the output value of a single sample test point. A comprehensive evaluation function is needed to synthesize the sensitivity analysis results of each single sample point. Here, we apply the MSA metrics proposed by Jacek. *M* [25] is chosen as the synthetic evaluation function. Assume S_{ik} denotes the sensitivity coefficient of the input variable *i* of all the samples to the output variable $Y_k(k=1)$, and s_{ik}^t denotes the sensitivity coefficient of input variable *i* to the output variable $Y_k(k=1)$ in the number *t* sample. The synthetic evaluation function can be expressed as (15):

$$S_{ik} = \sqrt{\frac{\sum_{t=1}^{n} \left(s_{ik}^{t} \right)^2}{n}}, \quad (15)$$

where n denotes the total number of samples, and S_{ik} is nonnegative. S_{ik} can be used to sort the sensitivity of the input bands, and then the influence of the input variable to the output results can be measured.

3.2.2 Band Classification Using Neural Network Sensitivity

First, according to the proportion Rs in each subspace, several bands are selected to form a band combination. Combined with the pre-selected object type, the training sample P and the test sample P_Test are generated, which then serve as the input variables of the BP neural network. The number of band combinations is equal to the number of the neurons at the input end. Furthermore, the label value of the ground object type is used as the output of the BP neural network. Then, the training samples (with results T) and test samples (with results T_Test) are generated. In addition, the weight and threshold of the BP neural network are optimized by a DE algorithm. All of the samples are classified by the optimized BP neural network, and the results of the sensitivity analysis are calculated. In addition, bands are sorted according to their sensitivity coefficients. Bands with smaller sensitivity coefficients will be given up, and bands with important effects on classification results will be identified. Last, the screened band combinations are classified using the neural network classifier to verify the effect of dimension reduction. The flow of the procedure is shown in Figure 1.

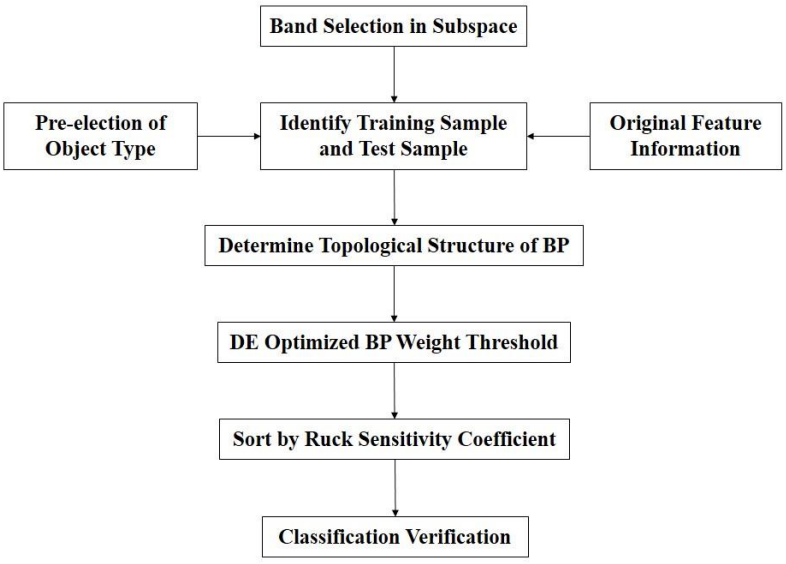


Fig. 1. The realization flow for band selection by sensitivity

3.2.3 Classification of Hyperspectral Remote Sensing Image based on Multi-instance Learning

As described above, in multi-instance learning [31–32], the training set is composed of labeled bags, and the bags are made up of non-labeled samples. The goal of multi-instance learning is to predict the label of previously unseen bags by learning the training set. Classification by bags as a unit includes many more features than classification by pixels or by the object as a unit, and both the anti-noise capability of the region and the accuracy of classification are improved. Each feature of the segmented object is taken as an instance; the object set generated by the clustering is taken as the packet. Using the packets, multi-instance learning based on an SVM is used for classification. In this paper, first, based on the dimension reduction of the band, the watershed transform algorithm is utilized to decompose the image into several objects that are used as instances for multi-instance learning. Then, the training samples are constructed with instances selected by artificial selection, and the unknown bags are obtained by the clustering algorithm. The new bag is marked with the nearest sample category that has maximum diversity. In other words, the number of generated bags is completely determined by the clustering algorithm. The steps of the algorithm are as follows.

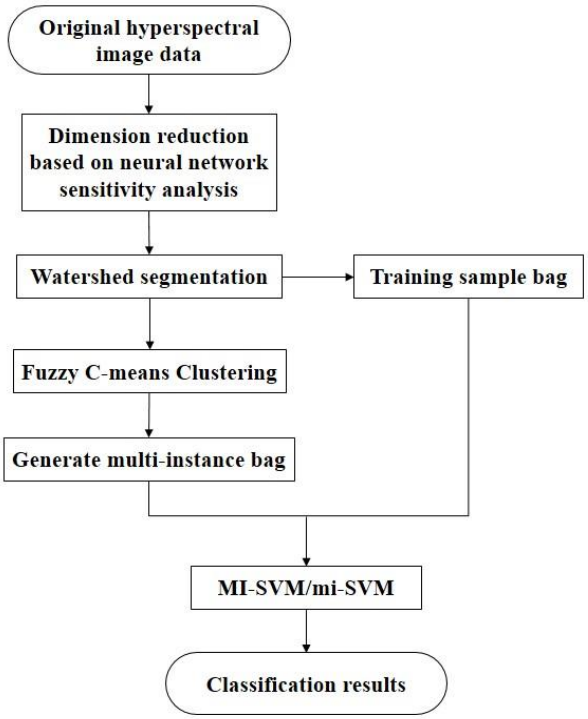


Fig. 2. The realization flow of classification of hyperspectral remote sensing images based on multi-instance learningž

(1) Reduce the dimension of the hyperspectral band by using neural network sensitivity analysis.
(2) Obtain the object by applying the common watershed algorithm.
(3) For each band that is selected by the band selection method, calculate the property of the object, and construct the property space.
(4) Multi-instance bags are generated by the fuzzy C-means clustering
(5) Select samples for each kind of object to form a training sample bag, and use an SVM for classification.
(6) Get the classification results.

The flow of the classification process is shown Figure 2.

4. Experiments and Results

4.1. Design of the Experiment

We designed some simulation experiments to demonstrate empirically the effectiveness of our proposed method for dimension reduction of hyperspectral remote sensing images using neural network sensitivity analysis and multi-instance learning. The experimental program was developed using MATLAB R2009b. The SVM classifier utilized the LIBSVM toolkit (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) and adopted a radial basis function (RBF) to perform the experiments. The penalty factor of the SVM was 16. The BP neural network was realized by the built-in neural network toolbox of MATLAB. We designed the experiment using a standard hyperspectral image. The dataset was revised in MAT format, which can be obtained from the website http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scens.
The image was part of a hyperspectral image from a mixed agroforestry experimental zone in northwestern Indiana (USA). It was acquired by the AVIRIS sensors in June 1992. In this image, the range of wavelengths was 0.4~2.5 um, the size of image was 145*145 pixels, and the spatial resolution of the image was 25m. In the pretreatment of the original image [24] (including image denoising and image deblurring), the bands that were heavily polluted by water vapor and noise (such as bands 1~4, 78, 80~86, 103~110, 149~165, and 217~224) were removed from the original bands. The remaining 179 bands were kept for experimental purposes. Figure 3 is an RGB false color image composite of the selected bands: 50, 27, and 17. Figure 4 is the real distribution image of the 7 objects chosen for the experiments.

There were 16 classes of ground objects in the image. Seven of these classes had larger sample sizes and were chosen for the classification experiments. The ratio between training samples and testing samples was selected evenly as 1:1. Table 1 displays the number of ground objects, as well as the names, and quantities of training and testing samples.

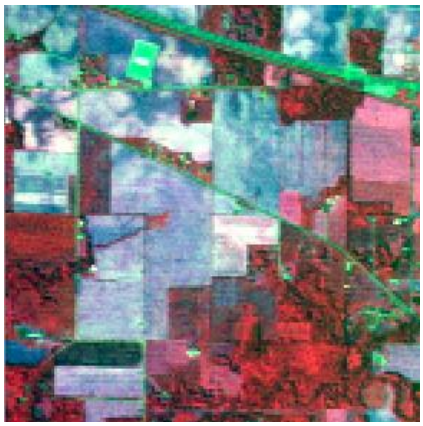


Fig. 3. The AVIRIS false color image

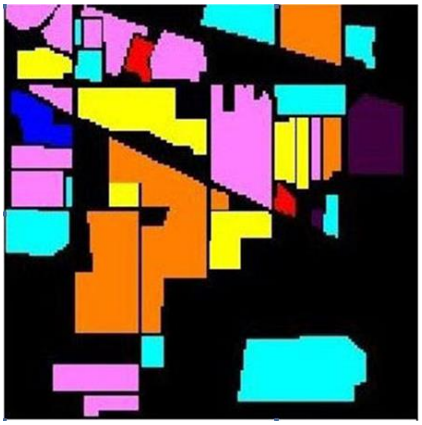


Fig. 4. The real distribution image

Table1. Training samples and test samples

Class number	Training samples	Training bags
Class 1	717	15
Class 2	417	9
Class 3	374	8
Class 4	484	10
Class 5	1234	25
Class 6	307	6
Class 7	647	13
Total	4180	86

4.2. Results and Analysis

A DE algorithm was used to optimize the BP neural network to get better results from the sensitivity analysis, which meant the band combination could improve the accuracy of classification effectively after the band with the smallest sensitivity coefficient was eliminated. Furthermore, to some extent, the multi-instance learning method based on the SVM could improve classification accuracy. We designed six experiments to demonstrate the superiority of the multi-instance learning method based on an SVM and optimizing the BP neural network with the DE algorithm. The band combinations used in the experiments were the same ones selected under the same subspace division. The detailed content of the experiments is shown in Table 2.

Table 2. Comparison experiments

Group	Content of Experiments
A	BP neural network classification (BP)
B	Sensitivity analysis and BP neural network classification (SA-BP)
C	Sensitivity analysis and Genetic algorithm optimizing BP neural network classification (SA-GABP)
D	Sensitivity analysis and Differential Evolution optimizing BP neural network classification (SA-DEBP)
E	Sensitivity analysis and MI-SVM (SA-MI-SVM)
F	Sensitivity analysis and mi-SVM (SA-mi-SVM)

In each experiment, six types of bands with different numbers were compared, and the values of *Rs* were selected relatively as 1/9, 1/6, 2/9, which meant selecting the band whose number was about 20, 30, or 40 from the divided subspace in accordance with the proportion of *Rs*. The topology structure of the BP neural network was set as the *N* number of neurons of the input layer equal to the number of bands in each group. The number of the types of primary features was 7, which was also the total number of the classification. The number of the neurons of output layer *M* was set as 7. The hidden layer was set to be single, and the number of its neurons *L* was set according to the expression $L=\sqrt{N+M}+a$. In this expression, *a* is the adjustment constant between 1 and 10. *L* changed with the training sample set, and ultimately it was determined that the error of the network was minimal when a = 5. The related parameters of the BP neural network training and DE algorithm were set as shown in Table 3 and Table 4.

Table 3. The setting of the parameters of BP

BP Parameters	Parameters setting
Frequency of training	1000
Minimum mean square error	0.01
Learning rate	0.1
Hidden layer activation function	Hyperbolic tangent function tansig

Output layer activation function	Linear function purelin
Training function	Levenberg-Marquadt Back propagation algorithm

Table 4. The setting of the parameters of DE

Parameters	**Parameter values**
Individual dimension D	D = N*L+L+L*M+M
Population size Nd	Nd = 20
Population size MAXGEN	MAXGEN = 50
Hybridization parameters CR	CR = 0.9
Differential evolution model	DE/best/1/bin

Except for group A, the other five groups of experiments needed to calculate the sensitivity coefficient. Considering the large number of sensitivity coefficients, Tables 5–7 show only the sensitivity coefficients of three kinds of bands. In the experiments with group D, these bands were analyzed using the Ruck method. In the table, the sensitivity coefficients are arranged in order from largest to smallest.

Table 5. Sensitivity coefficient of 20 Bands

No.	Band No.	Sensitivity coefficient	No.	Band No.	Sensitivity coefficient
1	117	1.0674	11	185	0.5610
2	43	0.9070	12	199	0.5477
3	140	0.8505	13	173	0.5102
4	57	0.7437	14	93	0.5099
5	5	0.7364	15	72	0.4385
6	102	0.6796	16	134	0.4076
7	68	0.6647	17	24	0.3984
8	212	0.6485	18	87	0.3313
9	37	0.6098	19	126	0.2332
10	15	0.5881	20	204	0.1486

Table 6. Sensitivity coefficient of 30 Bands

No.	Band No.	Sensitivity coefficient	No.	Band No.	Sensitivity coefficient
1	16	1.2224	16	141	0.3852
2	114	0.8933	17	73	0.3778
3	178	0.8743	18	37	0.3076
4	34	0.8259	19	87	0.3013
5	53	0.7306	20	117	0.2566

6	133	0.7176	21	148	0.2474
7	97	0.7028	22	191	0.1856
8	100	0.6639	23	200	0.1725
9	89	0.6369	24	170	0.1615
10	211	0.6300	25	8	0.1520
11	39	0.6259	26	42	0.1117
12	125	0.5951	27	11	0.0909
13	27	0.5242	28	57	0.0867
14	196	0.4252	29	68	0.0814
15	184	0.3990	30	166	0.0596

Table 7. Sensitivity coefficient of 40 Bands

No.	Band No.	Sensitivity coefficient	No.	Band No.	Sensitivity coefficient
1	16	1.3678	21	114	0.4461
2	68	1.1337	22	141	0.4298
3	23	1.1020	23	170	0.3795
4	120	0.9650	24	125	0.3689
5	196	0.9390	25	39	0.3623
6	133	0.8981	26	180	0.3424
7	27	0.8731	27	49	0.3112
8	89	0.8263	28	73	0.3083
9	166	0.8159	29	211	0.2794
10	11	0.7527	30	8	0.2504
11	117	0.7440	31	87	0.2373
12	138	0.7327	32	191	0.2334
13	214	0.6805	33	200	0.2145
14	37	0.6289	34	42	0.1798
15	148	0.6065	35	178	0.1409
16	34	0.5631	36	57	0.1275
17	102	0.5511	37	184	0.1172
18	100	0.5422	38	78	0.0988
19	147	0.5087	39	97	0.0827
20	53	0.4860	40	14	0.0408

From Tables 5–7, we can see that the sensitivity coefficient of the final band was small enough, about the 1/20 of the first one. For several experiments regarding classification, first each band whose sensitivity coefficient was less than 0.3 was excluded. Then, combined with multi-instance learning, the band combinations after dimension reduction were classified again by the BP neural network, checking whether the dimension reduction with sensitivity analysis was effective in improving the accuracy of classification. Table 8 shows the classification accuracy of band combinations after dimension reduction in the experiments of groups A, B, C, D, E, F.

Table 8. Classification accuracy

Number of Bands	A	B	C	D	E	F
20	82.75%	83.68%	83.75%	84.13%	92.43%	93.09%
30	83.75%	84.02%	84.73%	85.14%	92.78%	93.32%
40	84.90%	85.24%	85.52%	85.83%	93.18%	93.96%

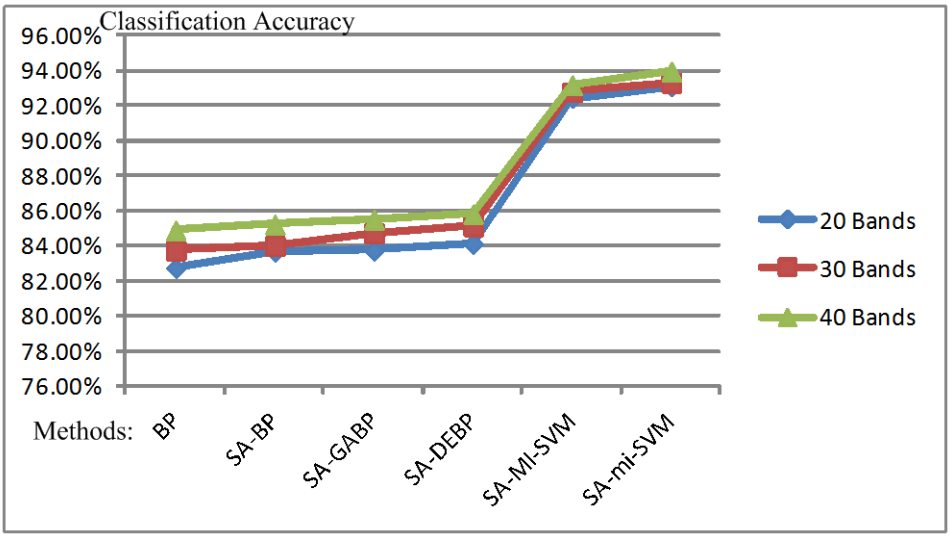
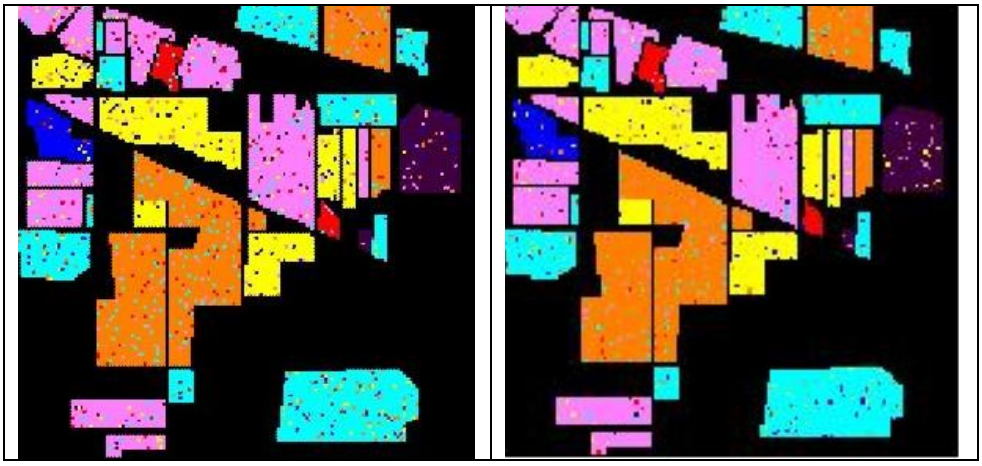


Fig. 5. Classification accuracy curves

From Table 8 and Figure 5, we note that with the same number of bands, the BP neural networks optimized with a genetic algorithm (GA) and DE algorithm in the experiments with groups C and D achieved better classification accuracy than groups A and B. This finding means that the sensitivity coefficients attained in the experiments with groups C and D provided more accurate reflection of the effects of each band on the classification results. Moreover, the BP neural network optimization with the DE algorithm in group D was better than for the GA algorithm adopted in group C. In the experiments with group B, the BP neural network was analyzed with the Ruck method after the division of the subspace without optimization, so the classification accuracy was lower than for groups C and D regardless of the number of bands. The BP neural network in experiments with group A was used in classification without dimension reduction, so its classification accuracy was the lowest. These results show that dimension reduction with sensitivity analysis was effective.

For classification, groups E and F used sensitivity analysis combined with multi-instance learning based on samples (mi-SVM) and multi-instance learning based on bags (MI-SVM), respectively. With the same number of bands, the classification accuracy of groups E and F was significantly higher than for groups B, C, and D, at more than 90%. These results demonstrate that in terms of classification, multi-instance learning performed better than the BP neural network. In addition, using multi-instance learning not only improved the classification accuracy, but also achieved an ideal classification effect. The classification accuracy using 30 bands was higher than when using 20 bands, and the classification accuracy using 40 bands was higher than when using 30 bands. This result indicates that with an increase in the number of bands, the classification accuracy of the same group improved. With the increased number of bands, the classification accuracy of groups E and F was more than 90%, which means that sensitivity analysis and multi-instance learning are very suitable for the classification of hyperspectral image. Figures 6–8 show the results of six groups of experiments using 20, 30, and 40 bands. It can be seen directly from these figures that groups A and B had the fewest incorrect classifications and higher classification accuracy regardless of the number of bands.

Finally, we compared the classification performance of the proposed method with some other hyperspectral image classification methods, as shown in Table 9. In this table, the accuracies outside the brackets were taken from corresponding references directly, and those accuracies in the brackets were obtained by our proposed method. All of the methods were tested under the same conditions with 7 classes and 5% trainings samples. Table 9 demonstrates that the classification performance of the proposed method outperformed all of the compared methods.



A B

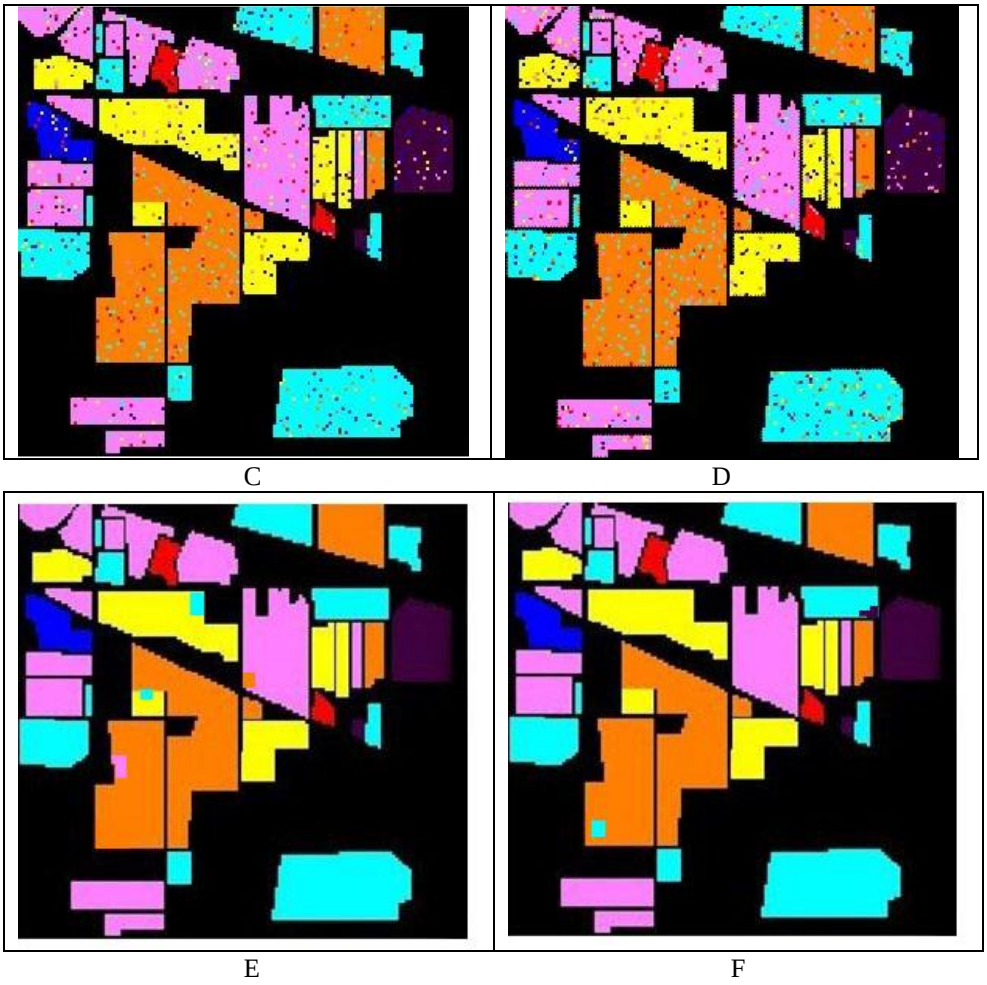


Fig. 6. Classification results of 20 bands

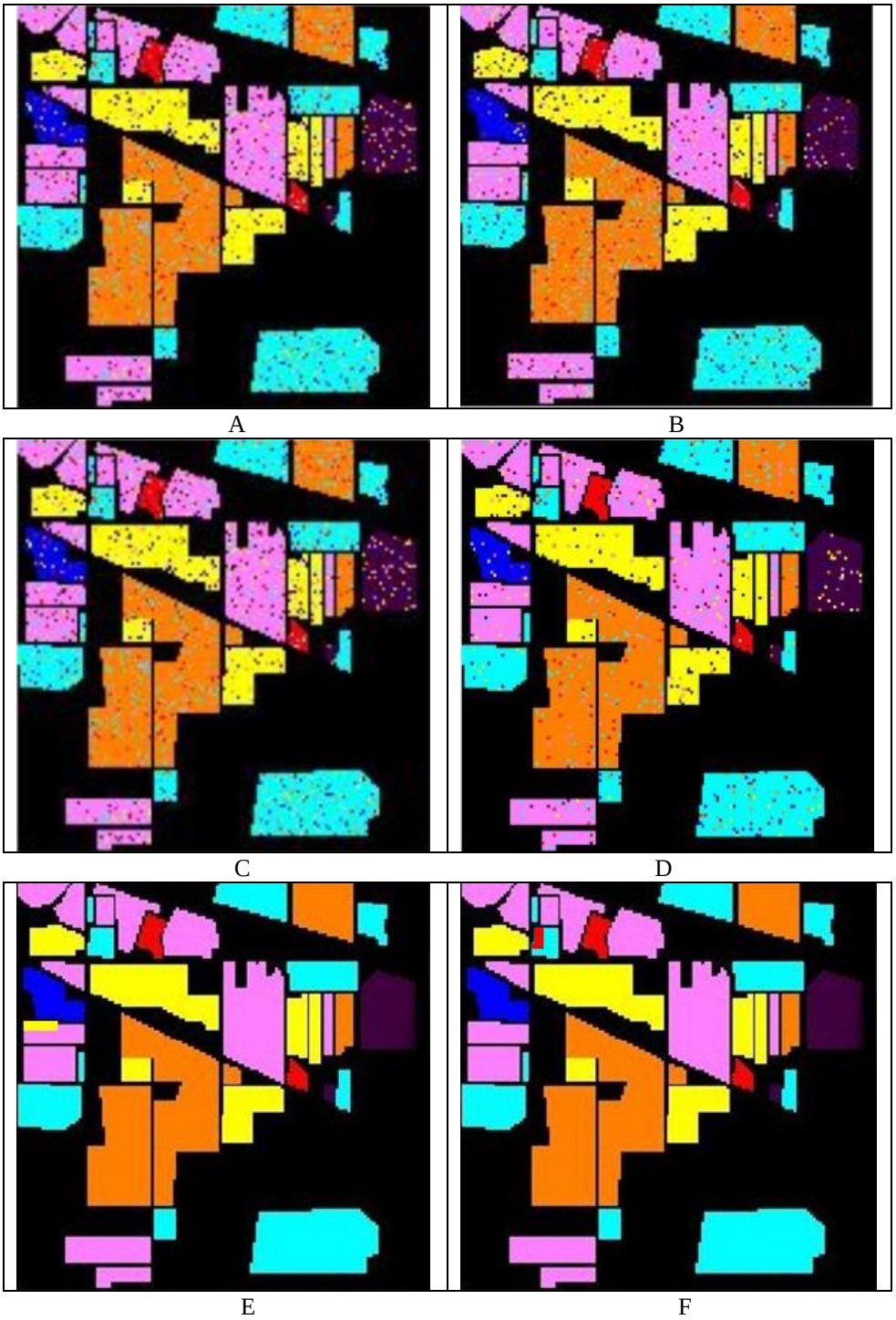


Fig. 7. Classification results of 30 bands

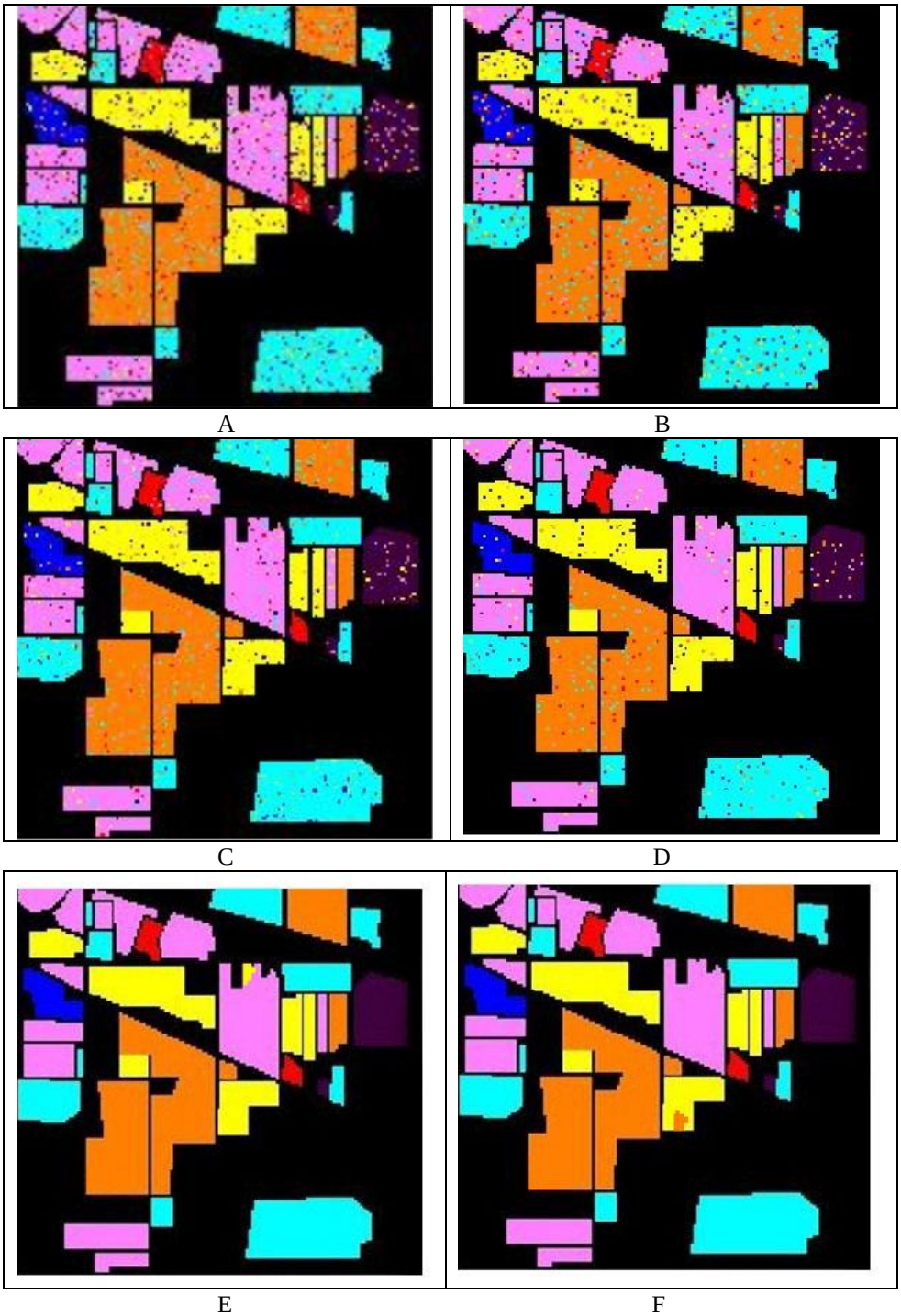


Fig. 8. Classification results of 40 bands

Table 9. Comparison with other methods

References	Baseline	Feature	Overall Accuracy
Sun et al. [33]	RWASL	Spectral	94.15%
Guo et al. [34]	FSAF	Spectral	84.36%
Li et al. [35]	KELM	Spectral	84.53%
Hu et al. [36]	RBF-SVM	Spectral	90.74%
Proposed method	SA-MI-SVM	Spectral	94.21%
Proposed method	SA-mi-SVM	Spectral	94.28%

5. Conclusions

Hyperspectral images take objects as the basic classification unit in a manner similar to the examples and packages used in multi-instance learning. Therefore, multi-instance learning can be used for the classification of remote sensing images. In this research, we combined neural network sensitivity analysis with a multi-instance learning algorithm based on an SVM to achieve dimension reduction and classification of hyperspectral remote sensing images. First, to reduce the correlation among the input properties, we used adaptive subspace division to select band combinations. In addition, a DE algorithm was adopted to optimize the BP neural network. To provide stable network connection weights and thresholds for sensitivity analysis by the neural network, we employed a sensitivity analysis method based on a partial derivative to calculate the sensitivity coefficient and remove bands that had very small coefficients. In this way, we achieved dimension reduction. To decrease the impact of the "different body with same spectrum" or "same body with different spectrum" phenomena on classification of hyperspectral images, a watershed transform algorithm was employed to decompose the image into several objects that were used as instances of multi-instance learning. The object set generated by clustering formed a bag, and the SVM was used for classification.

Our experimental results demonstrated that our method provided strong overall classification effectiveness when compared to prior methods. Based this study, we can draw the following conclusions.

1. Classification accuracy can be improved by means of the proposed neural network sensitivity analysis method because the contributions of bands for subsequent classification are sorted, and the bands with the largest contributions are selected contributions are selected.

2. A multi-instance learning method can learn robustly from highly noised training samples, but the bag label prediction rule for multi-instance learning also brings uncertainty to the classification results. The uncertainty problem can be controlled through the introduction of neural network sensitivity analysis.

3. The multi-instance learning algorithm based on an SVM can obtain higher classification accuracy under strong noise training conditions as long as the positive examples in the samples are selected properly.

4. For small sample training, the classification accuracy is higher based on the SVM and multi-instance learning algorithm.

Multi-instance learning has special advantages for resolving ambiguous problems compared with traditional supervised learning methods. Integrating multi-instance learning with neural network sensitivity analysis proved to have useful effects for controlling uncertainty. However, as the experiments showed, ensemble multi-instance learning needs a reasonable feature embedding scale factor. The main focus of future work should be on methods for implementing dynamically or self-adaptively chosen optimal scale factors.

Acknowledgement. This paper is supported by National Natural Science Foundation of China (No.61701166), Fundamental Research Funds for the Central Universities (No. 2015B26914), the Projects in the National Science & Technology Pillar Program during the Twelfth Five-year Plan Period (2015BAB07B01), and Science and Technology Project of Jiangxi Provincial Education Office (GJJ160625).

References

1. Z. Zhong, B. Fan, J. Duan, L. Wang, et al, "Discriminant Tensor Spectral-Spatial Feature Extraction for Hyperspectral Image Classification," IEEE Geoscience and Remote Sensing Letters, vol. 12, no. 5, pp. 1028-1032, 2017.
2. M. Xu, F. Xu, C. Huang, "Image restoration using majorization-minimizaiton algorithm based on generalized total variation," Journal of Image and Graphics, vol. 16, no. 7, pp. 1317-1325, 2011.
3. B. C. Kuo, H. H. Ho, C. H. Li, C. C. Hung, J. S. Taur, "A Kernel-Based Feature Selection Method for SVM With RBF Kernel for Hyperspectral Image Classification," IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 7, no. 1, pp. 317-326, 2013.
4. P. Gurram, H. Kwon, "Coalition game theory based feature subset selection for hyperspectral image classification," in lgarss IEEE International Geoscience and Remote Sensing Symposium, Canada, 2014, pp. 3446-3449.
5. L. Wang, Y. Zeng, T. Chen, "Back propagation neural network with adaptive differential evolution algorithm for time series forecasting," Expert Systems with Applications, vol. 42, no. 2, pp. 855-863, 2015.
6. N. H. Ly, Q. Du, J. E. Fowler, "Sparse graph-based discriminant analysis for hyperspectral imagery," IEEE Transactions on Geoscience and Remote Sensing, vol. 52, no. 7, pp. 3872-3884, 2013.
7. T. G. Dietterich, R. H. Lathrop, T. Lozano-Perez, "Solving the multiple-instance problem with axis-parallel rectangles," Artificial Intelligence, vol. 89, no. 1-2, pp. 31-71, 1997.
8. CHEN Jie, DENG Min, XIAO Peng-feng, YANG Min-hua, MEI Xiao-ming. Rough Set Theory Based Object-oriented Classification Of High Resolution Remotely Sensed Imagery. Journey of Remote Sensing, 2010, 14(6): 1139-1155.
9. Zhang C, Zhao Y J, Zhang D H, Zhao N B. Application and Evaluation of Object-oriented Technology in High-resolution Remote Sensing Image Classification. Control, Automation and Systems Engineering (CASE), 2011 International Conference on, 2011: 1-4.

10. YANG Chang-bao, DING Ji-hong. Study of Objected Based Remote Sensing Image Classification. [J]. Journal of Jilin University, 2006, 36(4): 642-646.
11. TAN Yu-min, HUAI Jian-zhu, TANG Zhong-shi. An Object-Oriented Remote Sensing Image Segmentation Approach Based On Edge Detection. Spectroscopy And Spectral Analysis, 2010, 30(6): 1624-1627.
12. Alimu Saimaiti, DU Pei-jun. Object Oriented High Resolution Remote Sensing Image Classification Based on Multi Instance Learning. Remote Sensing Information, 2012, 13(3): 60-66.
13. C. Yi, X. Yan, H. Dan, "On sensitivity analysis," Journal of Beijing Normal University (Natural Science), vol. 44, no. 1, pp. 9-16, 2008.
14. J. Zhang, Z. Q. Liu, H. Wang, "Susceptibility of landslide based on Artificial Neural Networks and fuzzy evaluating model," Science of Surveying and Mapping, vol. 37, no. 3, pp. 59-62, 2012.
15. GAO Hongmin, LI Chenming, ZHOU Hui et al. Dimension Reduction and Classification of Hyperspectral Remote Sensing Images Based on Sensitivity Analysis of Artificial Neural Network, Journal of Electronics & Information Technology, vol.38, no.11, pp. 2715-2723.
16. G. D. Garson, "Interpreting neural-network connection weights," AI Expert, vol. 6, no. 4, pp. 46-51, 1991.
17. T. Tchaban, M. J. Taylor, J. P. Griffin, "Establishing impact s of the inputs in a feedforward neural network," Neural Compute and Applications, vol. 7, no. 4, pp. 309-317, 1998.
18. Y. Dimopoulos, P. Bourret, S. Lek, "Use of some sensitivity criteria for choosing networks with good generalization ability," Neural Process Letters, vol. 2, no. 6, pp. 1-4, 1995.
19. D. W. Ruck, S. K. Rogers, M. Kabrisky, "Feature Selection Using a Multilayer Perceptron," Journal of Neural Network Computing, vol. 2, no. 2, pp. 40-48, 1990.
20. J. D. Olden, D. A. Jackson, "Illuminating the "black box": a randomization approach for understanding variable contributions in artificial neural networks," Ecological Modelling, vol. 154, no. 1/2, pp. 135-150, 2002.
21. M. Scardi, L. W. H. Jr, "Developing an empirical model of phytoplankton primary production: a neural network case study," Ecological Modelling, vol. 120, no. 2/3, pp. 213-223, 1999.
22. R. Liu, X. Zhang, L. Zhang, et al, "Bitterness intensity prediction of berberine hydrochloride using an electronic tongue and a GA-BP neural network," Experimental and Therapeutic Medicine, vol. 7, no. 6, pp. 1696-1702, 2014.
23. W. J. Qian, T. C. Li, L. Ding, "Sensitivity analysis of reservoir's seepage discharge based on improved BP network," Journal of China Three Gorges University (Natural Sciences), vol. 34, no. 6, pp. 23-27, 2012.
24. J. M. Zurada, A. Malinowski, S. Usui, "Perturbation method for deleting redundant inputs of perceptron networks," Neurocomputing, vol. 14, no. 2, pp. 177-193, 1997.
25. H. M. Gao, L. Z. Xu, C. M. Li, et al, "A new feature selection method for hyperspectral image classification based on simulated annealing genetic algorithm and Choquet fuzzy integral," Mathematical Problems in Engineering, no.5, pp. 1-13, 2013.
26. DU Peijun, SAMAT Alim, Multiple instance ensemble learning method for high-resolution remote sensing image classification, Journal of Remote Sensing, 2013, 17(1):77-86.
27. Z. Qi, Y. Tian, Y. Shi, "Multi-instance classification based on regularized multiple criteria linear programming," Neural Computing & Applications, vol. 23, no. 3-4, pp. 857-863, 2013.
28. S. Andrews, I. Tsochantaridis, T. Hofmann, "Support vector machines for multiple-instance learning," in Proc of Advances in Neural Information Processing Systems, Vancouver: MIT Press, 2002, pp. 561-568.
29. F. Yu, X. Xu, "A short-term load forecasting model of natural gas based on optimized genetic algorithm and improved BP neural network," Applied Energy, vol. 134, no. 134, pp. 102-113, 2014.

30. J. Zhang, Y. Zhang, B. Zou, T. Zhou, "Fusion Classification of Hyperspectral Image Based on Adaptive Subspace Decomposition," in IEEE Signal Processing Society, Canada, 2000, vol.3, pp. 472-475.
31. Y. Tarabalka, M. Fauvel, J. Chanussot, J. A. Benediktsson, "SVM- and MRF- based method for accurate classification of hyperspectral images," IEEE Geoscience and Remote Sensing Letters, vol. 7, no. 4, pp. 736-740, 2010.
32. D. Tuia, F. Ratle, F. Pacifici, M. F. Kanevski, W. J. Emery, "Active learning methods for remote sensing image classification," IEEE Transactions on Geoscience and Remote Sensing, vol. 47, no. 7, pp. 2218-2232, 2009.
33. Sun, B., Kang, X., Li, S., & Benediktsson, J. A. Random-walker-based collaborative learning for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing, 2017, 55(1), 212-222.
34. Guo, B., Shen, H. and Yang, M.,. Improving Hyperspectral Image Classification by Fusing Spectra and Absorption Features. IEEE Geoscience and Remote Sensing Letters, 2017, 14(8), pp.1363-1367.
35. Li, J., Du, Q., Li, W., Li, Y.S. Optimizing extreme learning machine for hyperspectral image classification. J. Appl. Remote Sens. 2015, 9, 097296.
36. Hu,W., Huang, Y.Y., Wei, L., Zhang, F., Li, H.C. Deep Convolutional Neural Networks for Hyperspectral Image Classification. J. Sens. 2015, 2015, 258619.

Hui Liu received his master's degrees in computer application technology from Jiangxi University Of Science And Technology, China, in 2010. He is currently a Ph.D. candidate in computer science at Hohai University, China. His major research interests include remote sensing image processing and network engineering.

Chenming Li is an associate professor and the deputy dean of College of Computer and Information, Hohai University, Nanjing, China. He received his B S, M S and PhD degree in computer application technology from Hohai University, Nanjing, China, in 1993, 2003 and 2010, respectively. He is a senior member of China Computer Federation and Chinese Institute of Electronic. His current research interest includes information processing systems and applications, system modelling and simulation.

Lizhong Xu is a professor of College of Computer and Information Engineering and the Director of Engineering Research Center of Telemetry, Remote Sensing and Information System, Hohai University, Nanjing, China. He received his PhD degree from China University of Mining and Technology, Xuzhou, China in 1997. He is a Senior Member of Chinese Institute of Electronic and China Computer Federation. Currently, his research area includes multi-sensor systems and information fusion, information processing systems and applications, signal processing in remote sensing and remote control, system modelling and optimization.

Received: April 28, 2018; Accepted: January 03, 2019

Computer Science and Information Systems 16(2):469–489 https://doi.org/10.2298/CSIS180601007L

Density-Based Clustering with Constraints

Piotr Lasek[1] and Jarek Gryz[2]

[1] University of Rzeszow, Poland
lasek@ur.edu.pl
[2] York University, Canada
jarek@cse.yorku.ca

Abstract. In this paper we present our *ic-NBC* and *ic-DBSCAN* algorithms for data clustering with constraints. The algorithms are based on density-based clustering algorithms *NBC* and *DBSCAN* but allow users to incorporate background knowledge into the process of clustering by means of instance constraints. The knowledge about anticipated groups can be applied by specifying the so-called *must-link* and *cannot-link* relationships between objects or points. These relationships are then incorporated into the clustering process. In the proposed algorithms this is achieved by properly merging resulting clusters and introducing a new notion of deferred points which are temporarily excluded from clustering and assigned to clusters based on their involvement in *cannot-link* relationships. To examine the algorithms, we have carried out a number of experiments. We used benchmark data sets and tested the efficiency and quality of the results. We have also measured the efficiency of the algorithms against their original versions. The experiments prove that the introduction of instance constraints improves the quality of both algorithms. The efficiency is only insignificantly reduced and is due to extra computation related to the introduced constraints.

Keywords: data mining, data clustering, semi-supervised clustering, clustering with constraints, instance-level constraints

1. Introduction

Clustering is a well-known and often used data mining technique. Its goal is to assign data objects (or points) to different clusters so that objects that are assigned to the same cluster are more similar to each other than to objects assigned to other clusters [10].

Clustering algorithms can operate on different types of data sources such as databases, graphs, text, multimedia, or on any other datasets containing objects that could be described by a set of features or relationships [2]. Performing a clustering task over a dataset can lead to a discovery of unknown yet interesting and useful patterns or trends in the dataset. Since clustering algorithms do not require any external knowledge as input (except certain parameters such as k in the *k-Means* algorithm), the process of clustering, in contrast to classification, is often referred to as an unsupervised learning. However, there has always been a natural need to incorporate already collected knowledge into algorithms to make them better both in terms of efficiency and quality of results. This need led to the construction of a new branch of clustering algorithms based on *constraints*. Constraint-based clustering algorithms utilize the fact, that in many applications, the domain knowledge in the form of, say, labeled objects is already known or could be easily specified

by domain experts. Moreover, in some cases such knowledge can be automatically detected. Initially, researchers focused on algorithms that incorporated pairwise constraints on cluster membership or learned distance metrics. Subsequent research was related to algorithms that used many other kinds of domain knowledge [5].

In [12] and [13] we presented the implementation of two neighborhood-based clustering algorithms *ic-NBC* and *ic-DBSCAN*. These two algorithms combined the well-known *NBC* [20] and *DBSCAN* [8] algorithms with two instance-level constraints, *must-link* and *cannot-link*. In this paper, we build upon our previous work. In particular, in Section 4, we provide a formal background behind the algorithms. The standard concepts used in *ic-NBC* and *ic-DBSCAN* (e.g. k-neighborhood, dense point, direct neighborhood-based density reachability, neighborhood-based density reachability, cluster, noise, nearest cluster, parent cluster, etc.) had to be adjusted to the new context of instance constraints and required new definitions. To improve readability, we have introduced a number of examples and figures illustrating the new concepts. Last but not least, we have added an entirely new section with experimental results to verify both quality as well as efficiency of the algorithms.

The paper is divided into six sections. In Section 2 we give a brief introduction to clustering with constraints and describe the related work in the field of constrained clustering – especially related to density-based clustering. In Section 3, the classic *DBSCAN* and *NBC* algorithms are described. In Section 4 we present our own method. Section 5 contains an experimental evaluation of our algorithms. Conclusions and further research is discussed in Section 6.

2. Constraints

2.1. Instance-level constraints

In clustering algorithms with constraints, background or expert knowledge can be incorporated into algorithms by means of different types of *constraints*. [5]. Several types of constraints have been identified so far, for example, instance constraints describing relations between objects or distance constraints such as inter–cluster δ–constraints and intra–cluster ϵ–constraints [2]. Nevertheless, the hard *instance-level* constraints seem to be most useful since the incorporation of just few constraints of this type can improve clustering accuracy. (We use the Silhouette score to measure clustering quality in our experiments.)

In [16] authors introduced two kinds of *instance-level* constraints, namely: the *must-link* and *cannot-link* constraints. These constraints are simple yet have interesting properties. For example *must-link* constraints are symmetrical, reflexive and transitive: if two points, p_0 and p_1 are in a *must-link* relationship, that is, $c_{=}(p_0, p_1)$ (see Table 1 for notation), then these points should be assigned to the same cluster. On the other hand, if two points r_0 and r_1 are in a *cannot-link* relationship, that is, $c_{\neq}(r_0, r_1)$, then these points must not be assigned to the same cluster.

Consider the following example based on Figure 1. In Figure 1.a we present a sample dataset with two must-link constraints $c_{=}(p_0, p_1)$ and $c_{=}(p_2, p_3)$. Each pair of points should be assigned to the same cluster.In Figure 1.b we present a sample dataset with one *cannot-link* constraint $c_{\neq}(p_0, p_1)$. The dataset should be clustered so that points p_0

Table 1. Notation related to *instance-level* constraints used in the paper and auxiliary variables used in pseudo-code of the algorithm.

Notation	Description
$C(p)$	The cluster to which a point p was assigned. If a point has not been decided yet to which cluster it should be assigned then $C(p)$ returns `UNCLASSIFIED`. If p is a *noise* point, then $C(p) =$ `NOISE`.
$C_{=}$	The set of pairs of points that are in a *must-link* relation.
$c_{=}(p_0, p_1)$	Two points p_0 and p_1 are in a *must-link* relation (must be assigned to the same resulting cluster).
$C_{=}(p)$	The set of points which are in a *must-link* relation with point p.
$C_{\neq}$	The set of pairs of points that are in a *cannot-link* relation.
$c_{\neq}(r_0, r_1)$	Two points r_0 and r_1 are in a *cannot-link* relation (must not be assigned to the same resulting cluster).
$C_{\neq}(r)$	The set of points which are in a *cannot-link* relation with point r.
$ClusterId$	The auxiliary integer variable used for storing currently-created clusters identifier.
$p.ClusterId$	By using such a notation we refer to a $ClusterId$ related to point p.
$p.ndf$	Such a notation is used to refer to a value of the *NDF* factor associated with point p.
R_d, R_t	The auxiliary variables for storing deferred points.
$DPSet$	The variable for storing dense points. It is used for in an iterative process of assigning points to clusters.

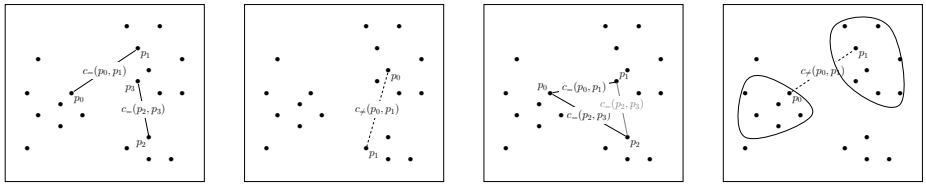
and p_1 will not be assigned to the same cluster.In Figure 1.c and Figure 1.d we illustrate basic features of instance constraints such as transitivity, reflexiveness, symmetry as well as entailment. □

2.2. Related Work

In constrained clustering algorithms, background or expert knowledge can be incorporated into algorithms by means of different types of constraints. Over the years, several methods of using constraints in clustering algorithms have been developed [5]. Constraint-based methods proposed so far employ techniques such as modifying the clustering objective function including a penalty for satisfying specified constraints [6], clustering using conditional distributions in an auxiliary space, enforcing all constraints to be satisfied during clustering process [17] or determining clusters and constraints based on neighborhoods derived from already available labelled examples [1]. In the distance-based methods, the distance measure is designed so that it satisfies given constraints [11,4]. Among algorithms proposed so far, a few represent modifications of density based algorithms, such as *C-DBSCAN* [15], *DBCCOM* [7] or *DBCluC* [18].

C-DBSCAN [15] is an example of a density-based algorithm using *instance-level* constraints where constraints are used to dictate whether some points may appear in the same cluster or not. In the first step, the algorithm partitions the dataset into subspaces using the *KD-Tree* [3] and then enforces *instance-level* constraints within each tree leaf producing so-called *local clusters*. Next, under *cannot-link* constraints, adjacent local clusters are merged enforcing *must-link* constraints. Finally, adjacent clusters are merged hierarchically enforcing remaining *cannot-link* constraints.

DBCluC [18] which was also based on the *DBSCAN* [8] employs an obstacle modelling approach for density-based clustering of large two–dimensional datasets. By means of the modelling it is also capable of detecting clusters of arbitrary shape and is not sensitive to the order of points in a dataset, constraints and noise. The efficiency of clustering



(a) **(b)** **(c)** **(d)**

Fig. 1. An illustration of (a) *must-link* constraints connecting points p_0 and p_1 as well as p_2 and p_3. Points that are connected by must-link constraint have to be assigned to the same cluster; (b) *cannot-link* constraint connecting points p_1 and p_2. In spite of the fact that points may be located relatively closely, if there is a *cannot-link* relation between them, they cannot by assigned to the same cluster; (c) transitive, reflexive and symmetrical features of *must-link* constraints. p_0 and p_1 as well as p_0 and p_2 are connected by *must-link* constraints, thus p_1 and p_2 are also connected by a *must-link* constraint; (d) entailed *cannot-link* constraints. All points from clusters s_0 and s_1 cannot be assigned to the same cluster because of $c_{\neq}(p_0, p_1)$ constraint

is leveraged by a reduction of polygons modelling the obstacles – the algorithm simply removes unnecessary edges from the polygons making the clustering faster in terms of number of constraints to be analysed. Nevertheless, the mechanism of obstacle reduction requires a complex preprocessing to be done before clustering.

The *DBCCOM* algorithm [7] pre-processes an input dataset by modeling the presence of physical obstacles - similarly to *DBCluC*. It also detects clusters of arbitrary shapes and size and is also considered to be insensitive to noise as well as an order of points in a dataset. The algorithm comprises of three steps: first, it reduces the obstacles by employing the edge reduction method, then performs the clustering and finally applies hierarchical clustering on formed clusters. The obstacles in the algorithm are represented as simple polygons and however the algorithm uses a more efficient polygon edge reduction algorithm than *DBCluC*. The results reported by the authors algorithm confirm that it can perform polygon reduction even faster than *DBCCOM* and can produce a hierarchy of clusters.

3. Density-based clustering

Density-based clustering algorithms use density functions to identify clusters. Clusters are dense regions separated by regions of empty space or low density called noise or outliers. Clusters generated in this way can be of arbitrary shape. In this section we describe two density-based algorithms: *DBSCAN* and *NBC*.

3.1. DBSCAN

The *DBSCAN* algorithm [8] is a well known density-based clustering algorithm. The algorithm takes three input parameters: D – the set of data points, ϵ – the radius of the

neighborhood, $MinPts$ – the minimal number of points within ϵ-neighborhood. Each point in D has an attribute called $ClusterId$ which stores the cluster's identifier and initially is equal to UNCLASSIFIED. The key definitions related to the *DBSCAN* algorithm shown below will be used in the sequel. Again, the general notation is given in Table 1.

Definition 1 (ϵ–neighborhood, or $\epsilon NN(p)$ of point p). *ϵ–neighborhood of point p is the set of all points q in dataset D that are distant from p by no more than ϵ; that is,*

$$\epsilon NN(p) = \{q \in D | dist(p,q) \leq \epsilon\},$$

where $dist$ is a distance function.

Clusters in *DBSCAN* are associated with core points which can be considered as seeds of clusters.

Definition 2 (core point). *p is a* core point *with respect to ϵ if its ϵ-neighborhood contains at least $MinPts$ point; that is, $|\epsilon NN(p)| \geq MinPts$.*

The point p_2 in Figure 3a is a core point as its ϵ–neighborhood contains 6 points (we assume $MinPts = 6$ in this case).

Definition 3 (directly density-reachable points). *Point p is directly density reachable from point q with respect to ϵ and $MinPts$ if the following two conditions are satisfied:*
a) $p \in \epsilon NN(q)$
b) *q is a core point.*

Figure 3a illustrates the concept of direct reachability.

Definition 4 (density-reachable points). *Point p is density-reachable from a point q with respect to ϵ and $MinPts$ if there is a sequence of points $p_1, ..., p_n$ such that $p_1 = q$, $p_n = p$ and p_{i+1} and is directly density-reachable from p_i, $i = 1 \ldots n-1$.*

Figure 3b illustrates the concept of reachability.

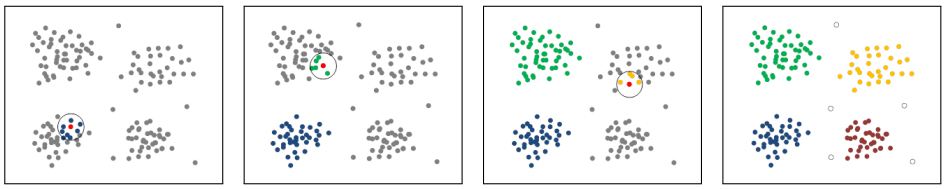
Definition 5 (cluster). *A cluster is a non-empty set of points in D which are density-reachable from the same core point.*

Although Definition 5 is formulated somewhat differently than the definition provided in [8], the resulting clusters are identical in both cases.

Points that are not in dense areas are not associated with any clusters and are considered noise.

Definition 6 (noise). *Noise is the set of all points in D that are not density-reachable from any core point.*

DBSCAN proceeds as follows. Firstly, the algorithm generates a label for the first cluster to be found. Next, the points in D are read. The value of the $ClusterId$ attribute of the first point read is equal to UNCLASSIFIED. While the algorithm analyzes point



(a) **(b)** **(c)** **(d)**

Fig. 2. Illustration of a sample execution of the *DBSCAN* algorithm. The neighborhood of the first core point is assigned to a cluster (a). Subsequent assignment of density-reachable points forms the first cluster; initial seeds are determined for the second cluster (b). The second cluster reaches its maximum size; the initial seeds are determined for the third cluster (c). The third cluster reaches its maximum size; the initial seeds are determined for the fourth cluster. Finally, *DBSCAN* labels noise points represented here by empty dots (d).

after point, it may happen that the $ClusterId$ attributes of some points may change before these points are actually analyzed. Such a case may occur when a point is density-reachable from a core point examined earlier. Such density-reachable points will be assigned to the cluster of a core point and will not be analyzed later. If a currently analyzed point p has not been classified yet (the value of its $ClusterId$ attribute is equal to `UNCLASSIFIED`), then the *ExpandCluster* function is called for this point. If p is a core point, then all points in $C(p)$ are assigned by the *ExpandCluster* function to the cluster with a label equal to the currently created cluster's label. Next, a new cluster label is generated by *DBSCAN*. Otherwise, if p is not a core point, the attribute $ClusterId$ of point p is set to `NOISE`, which means that point p will be tentatively treated as noise. After analyzing all points in D, each point's attribute $ClusterId$ stores a respective cluster label or its value is equal to `NOISE`. An illustration of a sample execution of *DBSCAN* has been ploted in Figure 2.

3.2. Neighborhood-based clustering

The Neighborhood-Based Clustering (*NBC*) [20] algorithm also belongs to the class of density based clustering algorithms. The characteristic feature of *NBC* compared to *DBSCAN* is its ability to measure relative local densities. Hence, it is capable of discovering clusters of different local densities and of arbitrary shape. The algorithm has two parameters: the set of points D and the number k which is used to describe density of a point.

The key definitions related to the *NBC* algorithm are presented below; k–neighborhood and k^+–neighborhood, defined below, are parameters used to describe dense neighborhoods.

Definition 7 (k–neighborhood, or $kNN(p)$). *k–neighborhood of point p is a set of k ($k > 0$) points satisfying the following conditions:*

$$|kNN(p)| = k, \text{ and}$$

$$\forall_{o' \in D \setminus kNN(p)} \forall_{o \in kNN(p)} \; dist(p, o' \geq dist(p, o).$$

Definition 8 (k^+–neighborhood, or $k^+NN(p)$). *k^+–neighborhood of point p is equal to $\epsilon' NN(p)$ where:*

$$\epsilon' = max(\{dist(p, v)|v \in kNN(p)\}).$$

k^+–neighborhood is similar to $\epsilon NN(p)$ (see Def. 3.1). However, ϵ is not a parameter given a priori to the algorithm, but a property of dense neighborhoods relative to a given data set.

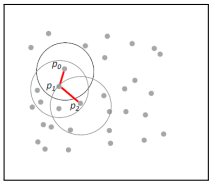
Definition 9 (punctured k^+–neighborhood). *Punctured k^+–neighborhood of point p $k^+NN(p^-)$ is equal to $k^+NN(p) \setminus \{p\}$; that is:*

$$k^+NN(p^-) = k^+NN(p) \setminus \{p\}.$$

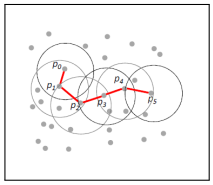
The concept of k^+–neighborhood of p is illustrated in Figure 4a.

Definition 10 (reversed punctured k^+–neighborhood of a point p). *Reversed punctured k^+–neighborhood of a point p $Rk^+NN(p)$ is the set of all points $q \neq p$ in dataset D such that $p \in k^+NN(q^-)$; that is:*

$$Rk^+NN(p) = \{q \in D|p \in k^+NN(q^-)\}.$$

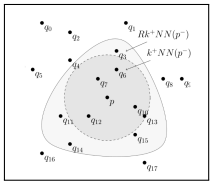


(a) p_0 is directly density-reachable from core point p_1; p_0 is density-reachable from p_2 ($MinPts = 6$).

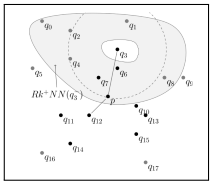


(b) Both p_0 and p_5 are density-reachable from core point p_2, so p_0 and p_5 belong to $C(p_2)$ ($MinPts = 6$).

Fig. 3. Illustration of some of the concepts of *DBSCAN*



(a) q_6 is directly neighborhood-based density reachable from p because $q_6 \in k^+NN(p^-)$.



(b) Point q_{12} is neighborhood-based density reachable from point q_3.

Fig. 4. Illustration of some of the concepts of *NBC* algorithm.

Definition 11 (neighborhood-based density factor of a point – $NDF(p)$). *Neighborhood-based density factor of a point p is defined as*

$$NDF(p) = |Rk^+NN(p)|/|k^+NN(p^-)|.$$

Points having the value of the value of NDF factor equal to or greater than 1, are considered dense.

Definition 12 (dense point). *Point p is called a* local dense point *if its $NDF(p)$ is greater than* 1.

Definition 13 (directly neighborhood–based density reachable). *A point p is directly neighborhood–based density reachable from point q if $p \in k^+NN(q^-)$ and q is a dense (core) point.*

Point q_6 in Figure 4a is directly neighborhood–based density reachable from point p.

Definition 14 (neighborhood-based density reachable). *A point p is neighbor-hood-based density reachable from r if p is directly neighborhood-based density reachable from q and r is directly neighborhood-based density reachable from q.*

Point q_{12} in Figure 4b is directly neighborhood–based density reachable from point q_3.

Definition 15 (cluster). *A cluster is a maximal non-empty subset of D such that for two points p and q in the cluster, p and q are neighborhood-based density-reachable from a local core point with respect to k, and if p belongs to cluster C and q is neighborhood-based density connected with p with respect to k, then q belongs to C.*

Definition 16 (noise). *Noise is the set of all points in D that do not belong to any cluster. In other words, noise is the set of all points in D that are not neighborhood-based density-reachable from any local core point.*

In order to find clusters, NBC starts with calculating values of NDF factors for each point p_i in a database D, $i = 0, 1, \ldots, |D|$. Next, for each p_i, a value of NDF is checked. If it is greater than or equal to 1, then p_i is assigned to the currently created cluster c (identified by the value of $ClusterId$). Next, the temporary variable $DPSet$ for storing references to points, is cleared and each point, say q, belonging to $k^+NN(p_i^-)$ is assigned to c. If $q.ndf$ is greater than or equal to 1, then q is also added to $DPSet$. Otherwise, q is omitted and a next point from $k^+NN(p_i^-)$ is analyzed. Further, for each point from $DPSet$, say r, $k^+NN(r^-)$ is computed. All unclassified points belonging to $k^+NN(r^-)$ are assigned to c and points having values of NDF greater than or equal to 1 are added to $DPSet$. Next, r is removed from $DPSet$. When $DPSet$ is empty, $ClusterId$ is incremented and a next point from D, namely p_{i+1}, is analyzed. Finally, if there are no more points in D having values of NDF factor greater than or equal to 1, then all unclassified points in D are marked as $NOISE$.

4. Clustering with Constraints

In this section we present two density-based algorithms with constraints based on *DBSCAN* and *NBC*. The main modification in both algorithms is the introduction of the `DEFERRED` points. The deferred points are in ϵ–neighborhood (for *DBSCAN* algorithm) or k^+–punctured neighborhood (for the *NBC* algorithm) of points involved in *cannot-link* relationship. The original algorithms are then first executed without the deferred points after which the points are assigned to appropriate clusters to satisfy their *cannot-link* constraints.

The *must-link* constraints are handled in a simple way. In the original algorithms, the construction of clusters originates from the core points. These points are kept in appropriate lists which are then updated in subsequent iterations of the algorithm. If a given core point p is involved in a *must-link* relationship with another core point r, then r is added

to the cluster originating in p. In this way, the algorithm can connect two remote regions via a bridge defined by the pair of points in a *must-link* relationship.

Our interpretation of the instance constraints is slightly different from most of the existing approaches which stop execution of the clustering algorithms upon the discovery of conflicting constraints. We believe that instance constraints do not necessarily have to be fully satisfied. *ic-NBC* and *ic-DBSCAN* use techniques similar to *DBCluC* [18] where the concept of so-called *obstacle points* was introduced. Obstacle points are ignored during the process of clustering. In our algorithms, we treat *cannot-link* constraints (along with their *nearest neighbors*) as points which constitute similar *obstacles*, but we do not ignore them completely during clustering

Thus, in the process of clustering, if a conflicting constraints exists, the algorithm does not have to be stopped, and conflicting points are labeled as `NOISE`.

4.1. ic-DBSCAN

In this subsection we offer a modified version of *DBSCAN* with constraints. First we introduce a definition of *deferred* points (Definition 17) and then present modified definitions of *cluster* and *noise* - Definition 18 and Definition 21, respectively.

Definition 17 (deferred point). *A point p is called* deferred *if it is in a* cannot-link *relationship with any other point or it belongs to a ϵ–neighborhood $\epsilon NN(q)$, where q is any point in a* cannot-link *relationship ($q \in C_{\neq}$). In the latter case we call q a parent point.*

Definition 18 (cluster). *A cluster is a maximal non-empty subset of D such that:*
- *for two* non-deferred *points p and q in the cluster, p and q are neighborhood-based density-reachable from a local core point with respect to k, and if p belongs to cluster C and q is also neighborhood-based density connected with p with respect to k, then q belongs to C;*
- *a* deferred *point p is assigned to a cluster C if the 1st-nearest punctured neighbour of p belongs to C ($1 - NN(p^{-}) \in C$), otherwise, p is considered as a noise point.*

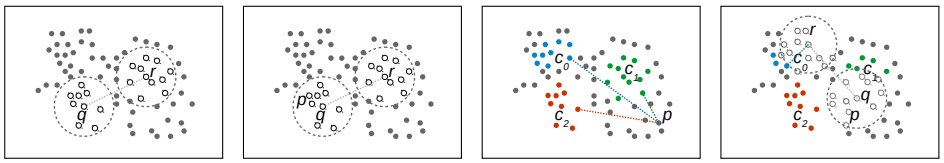
Definition 19 (nearest cluster). *A nearest cluster of a given point p is a cluster C to which p belongs.*

Definition 20 (parent cluster). *A parent cluster of a given point p (g_p) is a cluster C to which a parent point of p belongs.*

Definition 21 (noise). *The noise is the set of all points in D such that each of them is:*
- *not density-reachable from any core point or*
- *is a* deferred *point that has two or more neighbours at the same distance from it and thus can not be unambiguously assigned to a cluster.*

In other words, noise *is the set of all points in D that are not neighborhood-based density-reachable from any local* core point *and* deferred *points points that could not be assigned to any cluster.*

In the first phase, *ic-DBSCAN* algorithm (Figure 6a) omits all points which are involved in any *cannot-link* relationship and marks them as `DEFERRED`. Then, it adds those points to an auxiliary list called R_d which will be later used in the main loop of the algorithm using the *AssignDeferredPoints* function.



(a) Deferred points marked with empty circles.

(b) q is a parent point of point p.

(c) c_1 is a nearest cluster of point p.

(d) c_0 is a parent cluster of point p.

Fig. 5. An illustration of definitions of deferred points (a), parent point (b), nearest and parent cluster (c,d).

Then the algorithm iterates through all `UNCLASSIFIED` points from D except those which were added to R_d. For all of those points it calls the *ExpandCluster* function (Figure 6c) and passes all necessary parameters. The main modifications of the *ExpandCluster* function (compared to the classic *DBSCAN* algorithm) is in how the *must-link* constraints are processed. When a *must-link* point is processed and it is a *core point* or belongs to a neighbourhood of a point which is a *core point*, then it is assigned to $seeds$ or $curSeeds$ lists (containing *seed points*) depending on which part of the *ExpandCluster* function is currently executed. (The $seeds$ and $curSeeds$ are lists containing of points that belong to ϵ–neighborhoods of currently processed point in the *ExpandCluster* function and the number of points in the neighborhood is greater or equal to $MinPts$.)

The last part of the algorithm is to process the set of `DEFERRED` points. This is done by means of the *AssignDeferredPoints* function (Figure 6b). For each point q from R_d (a list of points which were marked as `DEFERRED` in the main algorithm method) the function determines what would be the parent cluster g_p of q. Next, it finds a point $p_{\neq}$ involved in *cannot-link* relationship and similarly determines its parent cluster $g_{p_{\neq}}$. Then, if those two parent clusters are the same ($g_p = g_{p_{\neq}}$) the `DEFERRED` point q cannot be assigned to the nearest cluster g_p and is labeled as `NOISE`. Otherwise, if two parent clusters are different q is assigned to g_p.

4.2. ic-NBC

In this subsection we offer our new neighborhood-based constrained clustering algorithm called *ic-NBC*. The algorithm is based on the *NBC* algorithm [20] but uses both *must-link* and *cannot-link* constraints for incorporating knowledge into the algorithm.

Below we present the definition of deferred point as well as modified definitions of cluster and noise - Definition 23 and Definition 24, respectively.

The *ic-NBC* algorithm employs the same definitions as *NBC* which are used in a process of clustering to assign points to appropriate clusters or mark them as noise. In *NBC* three types of points can be distinguished: *unclassified*, *classified* and *noise* points. In *ic-NBC*, we also employ a concept of `DEFERRED` points although defined slightly different than before.

Definition 22 (deferred point). *A point p is deferred if it is involved in a* cannot-link *relationship with any other point or it belongs to a k^+–punctured neighborhood $k^+NN(q^-)$, where q is any point involved in a* cannot-link *relationship.*

```
Algorihtm ic-DBSCAN(D, k, C=, C≠)
1.  R_d = ∅
2.  label all points in D as UNCLASSIFIED;
3.  ClusterId = label of a first cluster;
4.  for each point q involved in any constraint from C≠ do
5.      label q and points in ϵNN(q) as DEFERRED;
6.  endfor;
7.  add all DEFERRED points to R_d;
8.  foreach point p in set D \ R_d do
9.      if (p.ClusterId = UNCLASSIFIED) then
10.         if ExpandCluster(D, p, ClusterId, ϵ, MinPts) then
11.             ClusterId = NextId(ClusterId);
12.         endif;
13.     endif;
14. endfor;
15. AssignDefferedPoints(D, p, ClId, MinPts, ϵ, C=, C≠);
```

(a) The *ic-DBSCAN* algorithm.

```
Function AssignDeferredPoints(D, R_d, C≠)
1.  for each point q ∈ R_d do
2.      p ← GetParent(q);
3.      g_p ← NearestCluster(p);
4.      p≠ ← C≠(p);
5.      g_p≠ = NearestCluster(p≠);
6.      if g_p = g_p≠ then
7.          mark q as NOISE;
8.      else if
9.          assign point q to g_p;
10.     end if;
11.     remove q from R_d;
12. end for;
```

(b) Assigning deferred points to clusters.

```
Function ExpandCluster(D, p, ClId, MinPts, ϵ, C=, C≠)
1.  seeds = Neighborhood(D, p, ϵ);
2.  if |seeds| < MinPts then
3.      p.ClusterId = NOISE;
4.      return FALSE;
5.  else do
6.      for each point q in seeds do
7.          q.ClusterId = ClId;
8.          add C=(q) to seeds;
9.      endfor
10.     delete p from seeds;
11.     while |seeds| > 0 do
12.         curPoint = first point in seeds;
13.         curSeeds = Neighborhood(D, curPoint , ϵ);
14.         if |curSeeds| ≥ MinPts then
15.             for each point q in curSeeds do
16.                 add C=(q) to seeds;
17.                 if q.ClusterId = UNCLASSIFIED then
18.                     q.ClusterId = ClId;
19.                     append q to seeds;
20.                 else if q.ClusterId = NOISE then
21.                     q.ClusterId = ClId;
22.                 end if;
23.             end for;
24.         end if;
25.         delete curPoint from seeds;
26.     end while;
27. end else;
```

(c) The *ExpandCluster* function.

Fig. 6. The pseudo-code of the *ic-DBSCAN* algorithm using instance constraints.

Definition 23 (cluster). *A cluster is a maximal non-empty subset of D such that:*

- *for two* non-deferred *points p and q in the cluster, p and q are neighborhood-based density-reachable from a local core point with respect to k, and if p belongs to cluster C and q is also neighborhood-based density connected with p with respect to k, then q belongs to C;*
- *a* deferred *point p is assigned to a cluster C if the nearest neighbour of p which is not in* cannot-link *relationship with p belongs to C, otherwise p is considered as a noise point.*

Definition 24 (noise). *Noise is the set of all points in D that:*

- *have not been assigned to any cluster or*
- *each of them is a deferred point p whose $1^+NN(p^-)$ neighborhood contains points assigned to different clusters and thus can not be unambiguously assigned to a particular cluster.*

In other words, noise is the set of all points in D that are not neighborhood-based density-reachable from any local core point and deffered *points points which could be assigned to two or more clusters.*

ic-NBC (Figure 7a) can be divided into two main steps. In the first step the algorithm works very similarly to *NBC*. It calculates NDF factors and performs clustering. The main difference between *ic-NBC* and *NBC* is that the former:

```
Algorithm C-NBC(D, k, C=, C≠)
1.  R_d ← ∅
2.  ClusterId = 0;
3.  label all points in D as UNCLASSIFIED;
4.  CalcNDF(D, k);
5.  for each point q involved in any constraint from C= or C≠ do
6.      label q and points in k+NN(q−) as DEFERRED
7.      add q to R_d;
8.  endfor
9.  ClusterId = 0;
10. for each unclassified point p in D such that p.ndf ≥ 1 do
11.     p.ClusterId = ClusterId;
12.     clear DPSet;
13.     for each point q ∈ k+NN(p−) \R_d do
14.         q.ClusterId = ClusterId;
15.         if (q.ndf ≥ 1) then add q to DPset; endif
16.         add all points r from C=(q) such that
17.             r.ndf ≥ 1 to DPset;
18.     endfor
19.     while (DPSet ≠ ∅) do
20.         s = first point in DPset;
21.         for each unclassified point t in k+NN(s−) \R_d do
22.             t.ClusterId = ClusterId;
23.             if (t.ndf ≥ 1) then add t to DPset; endif
24.             add all points u from C=(t) such that
25.                 u.ndf ≥ 1 to DPset;
26.         endfor
27.         remove s from DPset;
28.     endwhile
29. ClusterId + +;
30. endfor
31. label unclassified points in D as NOISE;
32. AssignDeferredPointsToClusters(D, R_d, k, C≠);
```

(a) The *ic-NBC* algorithm.

```
Function AssignDefferedPointsToClusters(D, R_d, C≠)
Input:
    D - the input dataset (not clustered)
    R_d - the set of points marked as deferred
    k - the parameter of the C-NBC algorithm
    C≠ - the set of cannot-link constraints
Output:
    The clustered set with clusters satisfying cannot-link and
    must-link constraints.
1.  R_t ← R_d;
2.  do begin
3.      R_t ← R_d; // a temporary set for storing deferred points
4.                 // assigned to any cluster
5.      foreach point p in R_t do begin
6.          foreach point q in k+NN(p−) do
7.              if (q.ndf ≥ 1 and q.ClusterId > 0 and q ∈ R_d)
8.                  if (CanBeAssigned(p, q.ClusterId)) and
9.                      // checking if p can be assigned
10.                     // to a cluster identified by q.clusterId
11.                     (CanBeAssigned(
12.                         p.nearestCannotLinkPoint,
13.                         q.ClusterId)) then
14.                             p.ClusterId = q.ClusterId;
15.                             add p to R_a;
16.                     break;
17.                 endif
18.             endif
19.         endfor
20.         remove R_a from R_t;
21.     endfor
22. while (R_a ≠ ∅)
```

(b) The *AssignDefferedPointsToClustesrs* function.

Fig. 7. The pseudo-code of the *ic-NBC* algorithm.

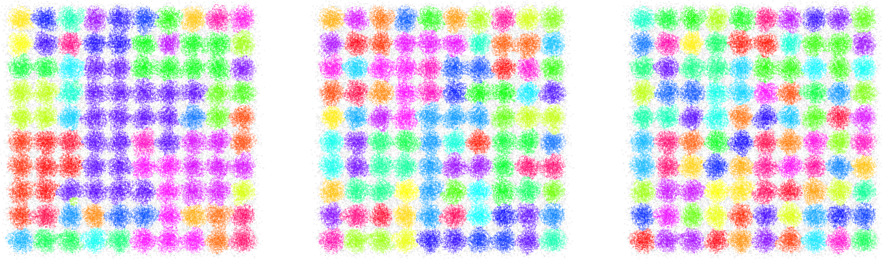
- determines which points will be considered as DEFERRED;
- excludes these points from all calculations (except to compute the values of NDF factors); and
- merges areas of clustered dataset according to *must-link* constraints.

The *ic-NBC* algorithm starts with the *CalcNDF* function. After calculating the NDF factors for each point from D, the deferred points are determined by scanning pairs of *cannot-link* connected points. These points are added to an auxiliary set R_d.

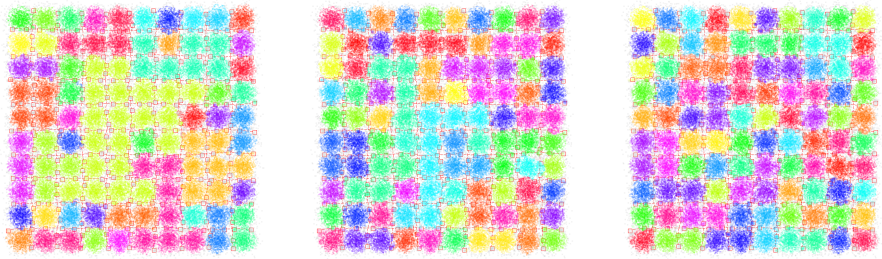
Then, the clustering process is performed in the following way: for each point p which was not marked as DEFERRED, it is checked if $p.ndf$ is less than 1. If $p.ndf < 1$, then p is omitted and a next from $DPSet$ is checked. If $p.ndf \geq 1$, then p as a *dense point* is assigned to the currently-created cluster identified by the current value of $ClusterId$.

Next, the temporary variable $DPSet$ is cleared and each non-deferred point, say q, belonging to $k^+NN(p^-) \setminus R_d$ is assigned to the currently-created cluster identified by the current value of the $ClusterId$ variable. Additionally, if $q.ndf \geq 1$, then it is assigned to $DPSet$ as well as all dense points which are in a must-link relation with q.

Next, for each unclassified point from $DPSet$, say s, its punctured k^+–neighborhood is determined. Each point, say t, which belongs to this neighborhood and has not been labeled as deferred yet is assigned to the currently created cluster and if its value of NDF is equal to or greater than 1, is added to $DPSet$. Moreover, all dense points which are in a must-link relation with t are added to $DPSet$ as well. Later, s is removed from $DPSet$



(a) k = 75, s = 0.063 **(b)** k = 100, s = 0.305 **(c)** k = 125, s = 0.434



(d) k = 75, s = 0.068 **(e)** k = 100, s = 0.303 **(f)** k = 125, s = 0.424

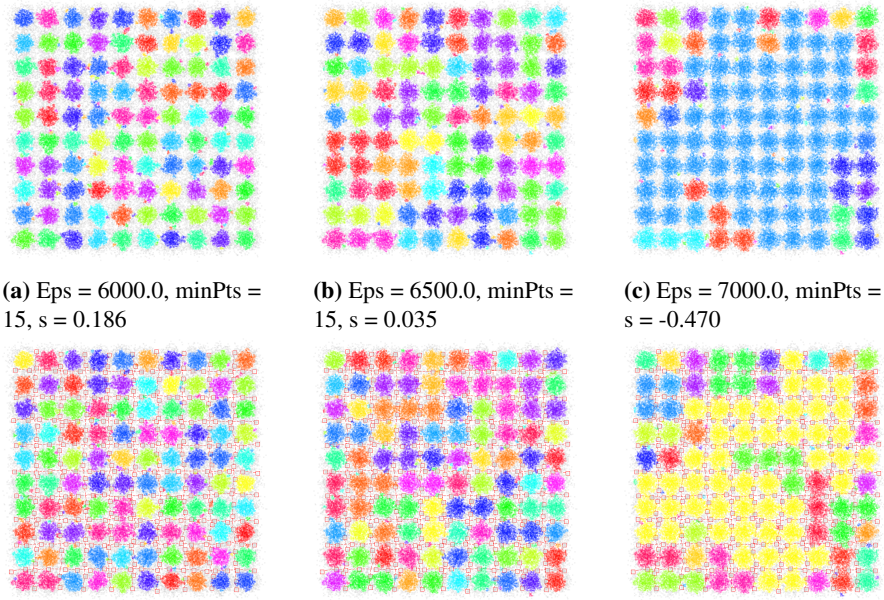
Fig. 8. Results of clustering for the Birch1 dataset using *NBC* (a-c), and *ic-NBC* (d-f). Colors represent mined clusters. k is a parameter if the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints.

and next point from $DPSet$ is processed. When $DPSet$ is emptied, then $ClusterId$ is incremented. After all points from D are processed, unclassified points are marked as noise by setting the values of their $ClusterId$ attribute to `NOISE`. However, in order to process the deferred points, the *AssignDeferredPointsToCluster* function is invoked. The function performs so that for each deferred point p it finds the nearest point q assigned to any cluster and checks whether it is possible (in accordance with cannot-link constraints) to assign p to the same cluster as q. Additionally, the function checks if the assignment of p to a specific cluster will not violate previous assignments of deferred points.

5. Experiments

In this section we present results of the experiments we performed to test the quality and efficiency of the proposed methods. We divided the experiments into two parts. First we focused on *quality* of clustering, then on the *efficiency*.

Datasets. For the experiments we used three standard two dimensional clustering benchmarking datasets (Birch) [19] with 100 000 points and 100 clusters. We examined three different versions of the Birch dataset containing clusters in regular grid structure (Birch1 - Figures 8-9), clusters at a sine curve (Birch2 - Figures 10-11) and random sized clusters in random locations (Birch3 - Figures 12-13).



(a) Eps = 6000.0, minPts = 15, s = 0.186

(b) Eps = 6500.0, minPts = 15, s = 0.035

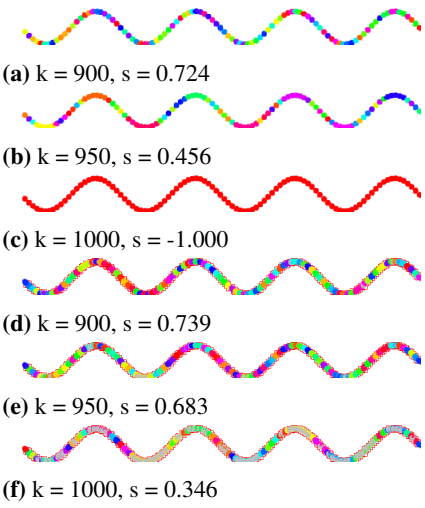
(c) Eps = 7000.0, minPts = 15, s = -0.470

(d) Eps = 6000.0, minPts = 15, s = 0.192

(e) Eps = 6500.0, minPts = 15, s = 0.077

(f) Eps = 7000.0, minPts = 15, s = -0.369

Fig. 9. Results of clustering for the Birch1 dataset using *DBSCAN* (a-c), and *ic-DBSCAN* (d-f). Colors represent mined clusters. Eps and $minPts$ are parameters of the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints.



(a) k = 900, s = 0.724

(b) k = 950, s = 0.456

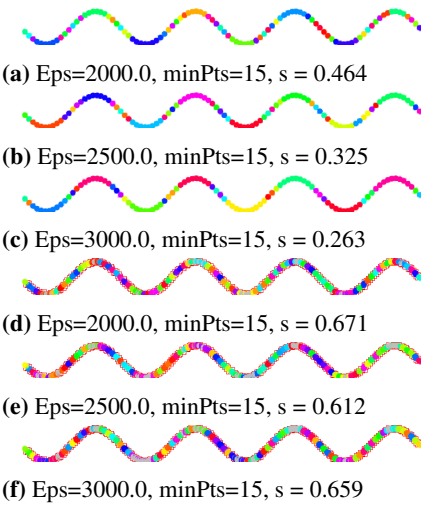
(c) k = 1000, s = -1.000

(d) k = 900, s = 0.739

(e) k = 950, s = 0.683

(f) k = 1000, s = 0.346

Fig. 10. Results of clustering for the Birch2 dataset using *NBC* (a-c), and *ic-NBC* (d-f). Colors represent mined clusters. k is a parameter if the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints.



(a) Eps=2000.0, minPts=15, s = 0.464

(b) Eps=2500.0, minPts=15, s = 0.325

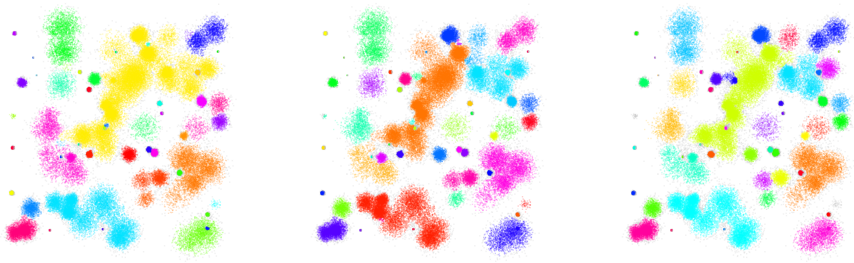
(c) Eps=3000.0, minPts=15, s = 0.263

(d) Eps=2000.0, minPts=15, s = 0.671

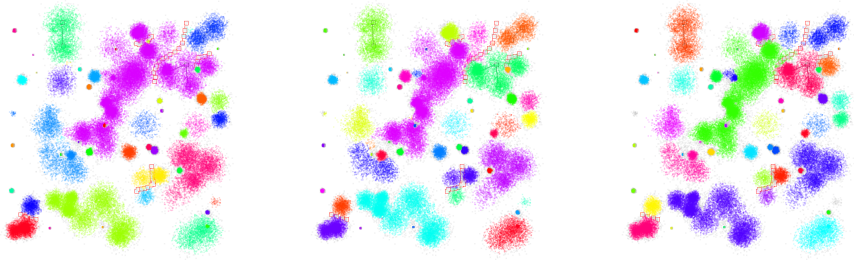
(e) Eps=2500.0, minPts=15, s = 0.612

(f) Eps=3000.0, minPts=15, s = 0.659

Fig. 11. Results of clustering for the Birch2 dataset using *DBSCAN* (a-c), and *ic-DBSCAN* (d-f). Colors represent mined clusters. Eps and $minPts$ are parameters of the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints.

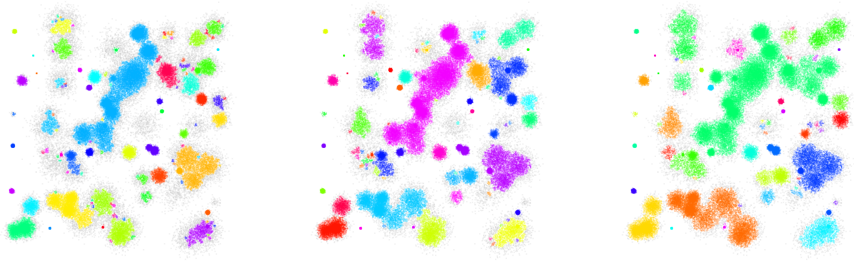


(a) k = 30, s = 0.107 **(b)** k = 50, s = 0.224 **(c)** k = 70, s = 0.282



(d) k = 30, s = 0.113 **(e)** k = 50, s = 0.240 **(f)** k = 70, s = 0.288

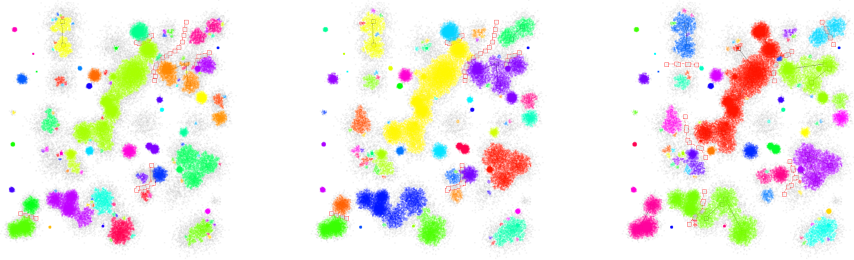
Fig. 12. Results of clustering for the Birch3 dataset using *NBC* (a-c), and *ic-NBC* (d-f). Colors represent mined clusters. k is a parameter if the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints. Black solid lines are *must-link* constraints. In the experiment 12 *must-link* and 43 *cannot-link* constraints were used.



(a) Eps = 5000.0, minPts = 15, s = 0.128

(b) Eps = 6000.0, minPts = 15, s = 0.146

(c) Eps = 7000.0, minPts = 15, s = 0.031

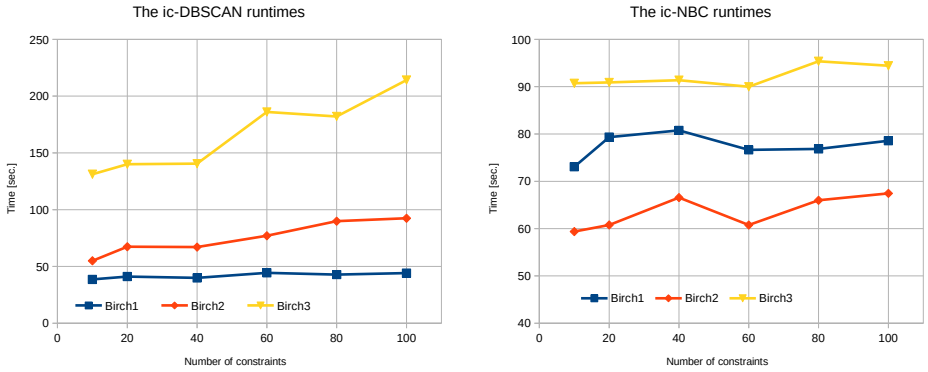


(d) Eps = 5000.0, minPts = 15, s = 0.154

(e) Eps = 6000.0, minPts = 15, s = 0.194

(f) Eps = 7000.0, minPts = 15, s = 0.171

Fig. 13. Results of clustering for the Birch3 dataset using *DBSCAN* (a-c), and *ic-DBSCAN* (d-f). Colors represent mined clusters. Eps and $minPts$ are parameters of the algorithm. s is a value of the Silhouette factor computed for the given clustering result. Red dashed lines denote *cannot-link* constraints. Black solid lines are *must-link* constraints. In the experiment 12 *must-link* and 43 *cannot-link* constraints were used.



(a) *ic-DBSCAN*.

(b) *ic-NBC*.

Fig. 14. Charts with the results of experiments for testing efficiency of *ic-DBSCAN* (a) and *ic-NBC* (b) for datasets Birch1, Birch2, and Birch3.

Table 2. Results of exeriments. Def - number of deferred points; Count - number of discovered clusters; Silh. - a value of the Silhouette score; Ind. - time of index building; Clust. - time of clustering; Tot. = Ind. + Clust. Times are given in milliseconds.

Data	Alg.	Param.	Ind.	Clus.	Def.	Cnt.	Tot.	Silh.
birch1	NBC	75	10145	63372		47	73517	0.063
birch1	ic-NBC	75	9006	67731	6537	49	76737	0.068
birch1	NBC	100	8980	61405		72	70385	0.305
birch1	ic-NBC	100	10399	74341	9877	73	84740	0.303
birch1	NBC	125	8934	67587		89	76521	0.434
birch1	ic-NBC	125	8791	80985	13542	89	89776	0.424
birch1	DBSCAN	6000.0, 15	9892	29946		223	39838	0.186
birch1	ic-DBSCAN	6000.0, 15	8879	34089	4518	216	42968	0.192
birch1	DBSCAN	6500.0, 15	9702	30969		152	40671	0.035
birch1	ic-DBSCAN	6500.0, 15	8885	34726	5386	146	43611	0.077
birch1	DBSCAN	7000.0, 15	9323	31151		73	40474	-0.470
birch1	ic-DBSCAN	7000.0, 15	9648	37155	6778	74	46803	-0.369
birch2	NBC	900	8955	108149		99	117104	0.724
birch2	ic-NBC	900	9260	187205	77742	100	196465	0.739
birch2	NBC	950	9011	118442		64	127453	0.456
birch2	ic-NBC	950	10281	210245	88899	92	220526	0.683
birch2	NBC	1000	9446	120449		1	129895	-1.000
birch2	ic-NBC	1000	9402	205044	87771	43	214446	0.346
birch2	DBSCAN	2000.0, 15	9370	46691		63	56061	0.464
birch2	ic-DBSCAN	2000.0, 15	9026	68285	26811	89	77311	0.671
birch2	DBSCAN	2500.0, 15	8807	51681		46	60488	0.325
birch2	ic-DBSCAN	2500.0, 15	9470	98568	55157	82	108038	0.612
birch2	DBSCAN	3000.0, 15	9243	56087		39	65330	0.263
birch2	ic-DBSCAN	3000.0, 15	8991	144539	101349	85	153530	0.659
birch3	NBC	30	9093	71470		52	80563	0.111
birch3	ic-NBC	30	8848	72752	213	48	81600	0.090
birch3	NBC	50	8712	78941		54	87653	0.234
birch3	ic-NBC	50	9984	79111	450	50	89095	0.149
birch3	NBC	70	8871	84654		50	93525	0.282
birch3	ic-NBC	70	10469	101585	622	45	112054	0.199
birch3	DBSCAN	5000.0, 15	9132	61933		118	71065	0.133
birch3	ic-DBSCAN	5000.0, 15	8482	63644	247	111	72126	-0.014
birch3	DBSCAN	6000.0, 15	9136	62968		80	72104	0.153
birch3	ic-DBSCAN	6000.0, 15	8644	72144	316	78	80788	0.020
birch3	DBSCAN	7000.0, 15	9293	67478		57	76771	0.031
birch3	ic-DBSCAN	7000.0, 15	8464	72594	515	57	81058	0.069

(a) Results of experiments designed to examine quality of proposed constrained algorithms compared to the original versions of the algorithms using the Silhouette score.

Data	Alg.	Param.	Cons.	Cnt.	Ind.	Clus.	Def.	Tot.
birch1	ic-NBC	100	10	62	8728	63990	359	73077
birch1	ic-NBC	100	20	55	9915	68614	803	79332
birch1	ic-NBC	100	40	39	9089	70127	1544	80760
birch1	ic-NBC	100	60	27	8896	65691	2054	76641
birch1	ic-NBC	100	80	15	9184	65275	2396	76855
birch1	ic-NBC	100	100	10	9746	66166	2661	78573
birch1	ic-DBSCAN	5000.0, 15	10	337	9319	28958	309	38586
birch1	ic-DBSCAN	5000.0, 15	20	337	9520	30872	712	41104
birch1	ic-DBSCAN	5000.0, 15	40	333	8739	29988	1192	39919
birch1	ic-DBSCAN	5000.0, 15	60	335	10501	31883	2002	44386
birch1	ic-DBSCAN	5000.0, 15	80	332	8738	31548	2533	42819
birch1	ic-DBSCAN	5000.0, 15	100	328	8960	31973	3216	44149
birch2	ic-NBC	50	10	92	9041	50207	117	59365
birch2	ic-NBC	50	20	81	8966	51558	237	60761
birch2	ic-NBC	50	40	65	10092	55825	633	66550
birch2	ic-NBC	50	60	49	8889	51154	708	60751
birch2	ic-NBC	50	80	32	9199	55838	942	65979
birch2	ic-NBC	50	100	22	9600	56700	1135	67435
birch2	ic-DBSCAN	1000.0, 15	10	94	9652	41497	3836	54985
birch2	ic-DBSCAN	1000.0, 15	20	82	11627	49121	6593	67341
birch2	ic-DBSCAN	1000.0, 15	40	68	9514	44713	12791	67018
birch2	ic-DBSCAN	1000.0, 15	60	53	9078	49770	18159	77007
birch2	ic-DBSCAN	1000.0, 15	80	36	9360	56373	24097	89830
birch2	ic-DBSCAN	1000.0, 15	100	39	9150	56351	26937	92438
birch3	ic-NBC	50	10	45	9050	81492	171	90713
birch3	ic-NBC	50	20	36	9494	80983	421	90898
birch3	ic-NBC	50	40	29	9165	81561	643	91369
birch3	ic-NBC	50	60	22	8499	80386	1097	89982
birch3	ic-NBC	50	80	15	10239	83934	1196	95369
birch3	ic-NBC	50	100	16	9206	83578	1627	94411
birch3	ic-DBSCAN	6000.0, 15	10	72	8711	90199	32349	131259
birch3	ic-DBSCAN	6000.0, 15	20	67	8621	90319	41090	140030
birch3	ic-DBSCAN	6000.0, 15	40	65	8433	91516	40586	140535
birch3	ic-DBSCAN	6000.0, 15	60	61	8815	109920	67327	186062
birch3	ic-DBSCAN	6000.0, 15	80	62	8760	107802	65545	182107
birch3	ic-DBSCAN	6000.0, 15	100	58	9075	124772	80327	214174

(b) Results of experiments testing the efficiency of the proposed constrained density based algorithms with respect to the number of constraints and values of the algorithms' parameters.

Implementation. Both implementations of the algorithms employ the same index structure – the *R-Tree* [9]. We implemented them in Java and performed the experiments on MacBook Pro 2.8GHz eight-core Intel Core i7, 16GB RAM. The source code can be found under the following link: `http://github.com/piotrlasek/clustering`

Quality. To examine how quality of clustering could be improved by means of instance constraints, we used the Silhouette score [14], a method of interpretation and validation of consistency within clusters. The *Silhouette score* for a point i is given by the following formula $s(i) = \frac{(b(i)-a(i))}{max\{a(i),b(i)\}}$, where $a(i)$ is the avarage dissimilarity of i with all other points within the same cluster, $b(i)$ is the lowest average dissimilarity of i to any other cluster to which i does not belong. The silhouette value measures *cohesion* and *separation* that means of how similar an object is to its own cluster compared to other cluster. The values of the silhouette score can range from -1 to $+1$, where a higher value indicates that the object was correctly assigned to its cluster. We report the mean Silhouette value over all objects in a dataset. Times and values of the Silhouette score are reported in Table 2a and Figures 8-14.

The introduction of instance constraints improves the quality of both *DBSCAN* as well as *NBC*; in most cases, the improvement is substantial. However, the clustering quality rises much more for *DBSCAN* than for *NBC*. *NBC* is designed - contrary to *DBSCAN* - to discover clusters with varying local densities (thanks to how the NDF factor was defined). In other words, *DBSCAN* mines clusters based on a global notion of density, *NBC* determines clusters using density calculated locally. For this reason, we do not see as much improvement in employing constraints in *NBC* compared to *DBSCAN*.

Efficiency. In the second part of the experiments we focused on time efficiency of clustering with respect to the number of constraints as well as values of algorithms' parameters (Table 2b, Figures 14a-b).

When performing experiments using *ic-NBC* we were changing the number of *must-link* and *cannot-link* constraints from 10 to 100. Since the additional operations must have been be performed in order to take the constraints into account, this was obvious that constrained versions of the algorithms had to be less effective than the original ones. However, as plotted in Figures 14a-b, the algorithms' execution times are almost constant with respect to the number of constraints used.

6. Conclusions

In this paper we have presented two clustering algorithms with constraints, *ic-NBC* and *ic-DBSCAN*, which were designed to let users introduce instance constraints for specifying background knowledge about the anticipated groups. In our approach we treat must-link constraints as more important than *cannot-link* constraints. Thus, we try to satisfy all must-link constraints (assuming, of course, that all of them are valid) before incorporating any cannot-link constraints. When processing *cannot-link* constraints, points which are contradictory (in terms of satisfying both *must-link* and *cannot-link* constraints) are marked as noise.

We have performed a number of experiments to test the quality of clustering and the efficiency of our algorithms by comparing them to their original versions. The results of the experiments clearly show that constraints are useful in clustering.

The experiments proved that the introduction of instance constraints improved the quality of clustering in both cases. At the same time, due to additional computations needed to process constraints, the performance of the algorithms was reduced but only marginally. The experiments also showed that the number of constraints does not have a critical impact on the algorithms performance.

In this work we have focused on of incorporating *instance-level* constraints into clustering algorithms by modifying the algorithms. Nevertheless, there are other ways of incorporating constraints into the process of clustering. For example, the constraints can be used to modify a distance matrix so that it reflects *must-link* and *cannot-link* relationships. Such a matrix can then be used as an input to the original algorithm without constraints. We believe that this is a promising area of research and we plan to explore it in future.

References

1. Basu, S., Banerjee, A., Mooney, R.: Semi-supervised clustering by seeding. In: In Proceedings of 19th International Conference on Machine Learning (ICML-2002. Citeseer (2002)
2. Basu, S., Davidson, I., Wagstaff, K.: Constrained clustering: Advances in algorithms, theory, and applications. CRC Press (2008)
3. Bentley, J.L.: Multidimensional binary search trees used for associative searching pp. 509–517 (1975)
4. Chang, H., Yeung, D.Y.: Locally linear metric adaptation for semi-supervised clustering. In: Proceedings of the twenty-first international conference on Machine learning, p. 20. ACM (2004)
5. Davidson, I., Basu, S.: A survey of clustering with instance level constraints. ACM Transactions on Knowledge Discovery from Data **1**, 1–41 (2007)
6. Davidson, I., Ravi, S.: Clustering with constraints: Feasibility issues and the k-means algorithm. In: SDM, vol. 5, pp. 201–211. SIAM (2005)
7. Duhan, N., Sharma, A.: Dbccom: Density based clustering with constraints and obstacle modeling. In: Contemporary Computing, pp. 212–228. Springer (2011)
8. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd, vol. 96, pp. 226–231 (1996)
9. Guttman, A.: R-trees: a dynamic index structure for spatial searching, vol. 14. ACM (1984)
10. Han, J., Kamber, M.: Data Mining, Southeast Asia Edition: Concepts and Techniques. Morgan kaufmann (2006)
11. Hertz, T., Bar-Hillel, A., Weinshall, D.: Boosting margin based distance functions for clustering. In: Proceedings of the twenty-first international conference on Machine learning, p. 50. ACM (2004)
12. Lasek, P.: C-nbc: Neighborhood-based clustering with constraints. In: Proceedings of the 23th International Workshop on Concurrency, Specification and Programming, vol. 1269, pp. 113–120 (2014)
13. Lasek, P.: Instance-level constraints in density-based clustering. In: Proceedings of the 24th International Workshop on Concurrency, Specification and Programming, pp. 11–18 (2015)
14. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics **20**(Supplement C), 53 – 65 (1987). DOI https://doi.org/10.1016/0377-0427(87)90125-7. URL `http://www.sciencedirect.com/science/article/pii/0377042787901257`
15. Ruiz, C., Spiliopoulou, M., Menasalvas, E.: C-dbscan: Density-based clustering with constraints. In: Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, pp. 216–223. Springer (2007)

16. Wagstaff, K., Cardie, C.: Clustering with instance-level constraints. AAAI/IAAI **1097** (2000)
17. Wagstaff, K., Cardie, C., Rogers, S., Schrödl, S., et al.: Constrained k-means clustering with background knowledge. In: ICML, vol. 1, pp. 577–584 (2001)
18. Zaïane, O.R., Lee, C.H.: Clustering spatial data when facing physical constraints. In: Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on, pp. 737–740. IEEE (2002)
19. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: A new data clustering algorithm and its applications. Data Mining and Knowledge Discovery **1**(2), 141–182 (1997)
20. Zhou, S., Zhao, Y., Guan, J., Huang, J.: A neighborhood-based clustering algorithm. In: Advances in Knowledge Discovery and Data Mining, pp. 361–371. Springer (2005)

Piotr Lasek received his M.Sc. degree in computer science from the Warsaw University of Technology in 2004 and the Ph.D. degree from the same university in 2012. Currently, is an Assistant Professor at the Rzeszow University, Poland. His main areas of research include interactive data mining and visualization.

Jarek Gryz is a Professor at the Department of Computer Science and Engineering at York University, Canada. He received his Ph.D. in Computer Science from the University of Maryland, USA, in 1997. His main areas of research involve database systems, data mining, query optimization via data mining, preference queries, query sampling.

Received: June 1, 2018; Accepted: February 20, 2019.

Computer Science and Information Systems 16(2):491–514 https://doi.org/10.2298/CSIS190122008J

Logical Filter Approach for Early Stage Cyber-Attack Detection

Vacius Jusas [1], Saulius Japertas [2], Tautvydas Baksys [3], Sandeepak Bhandari [4]

1 Software Engineering Department, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania
2 Department of Electronics Engineering, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania
3 Department of Computer Science, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania
4 Software Engineering Department, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania

Abstract. The planned in advance cyber-attacks cause the most damage for the users of the information systems. Such attacks can take a very long time, require considerable financial and human resources, and therefore, they can only be organized by large interest groups. Furthermore, current intrusion detection systems, intrusion prevention systems and intrusion response systems used to protect against cyber-attacks have several shortcomings. Such systems respond only to the attack itself when it is too late to take a preventive action and they are not suitable for detecting an attack in early stages when it is possible to block the attack and minimize the losses. Early detection requires detailed monitoring of network and system parameters to be able to accurately identify the early stages of the attack when it is still possible to kill the attack chain. In this paper, we propose to consider an attack chain consisting of nine stages. The method to detect early stage cyber-attack based on the attack chain analysis using hardware implementation of logical filters is suggested. The performed experiment acknowledges the possibility to detect the attack in the early stages.

Keywords: System security; Cyber-attack; Intrusion detection; Logical circuits.

1. Introduction

The most dangerous cyber-attacks are those that are planned in advance, and they can be planned by both state structures and terrorist organizations. The planned cyber-attacks consist of a variety of different stages. Different authors describe the different number of the stages and parameters of such cyber-attacks. Symantec entitles five stages: reconnaissance, incursion, discovery, capture, and exfiltration [1]. The same number of stages, but with different names, is proposed in [2]: reconnaissance, intrusion, taking control, collecting and leaking information, eliminating traces. Meanwhile, Yadav and Rao [3] offer seven steps: reconnaissance, weaponization, delivery, exploitation, installation, command and control, act on objective. Yadav and Rao [3] clearly distinguished two groups of the stages (early stages and late stages). More research works [4]-[6] can be found, where a number of stages varies between three and eight. Different means and equipment are used to organize the detection of the attacks at the different

stages. It can be assumed that detecting and stopping the attack in the early stages can prevent the serious harmful effects [4, 5]. However, it is necessary to distinguish between the early stages and the late stages, when damage created by an attack is mainly unavoidable. Therefore, it is needful to determine the various stages of attack in order to provide the means and methods for preventing the harmful effects of the attacks.

The aim of the paper is to present a method to determine the possibility of cyber-attack against information and telecommunication systems at its earliest stages when the cyber-attack can still be effectively stopped. The method is based on the detailed monitoring of network and system parameters to accurately identify the early stages of the attack. A hardware implementation of logical filters is suggested for the method.

The rest of the paper is organized as follows. We review the related work in Section 2. We present an attack vector of early stages in Section 3. We introduce early stage cyber-attacks detection method in Section 4. We discuss the results of the experiment in Section 5. We finish with conclusions in Section 6.

2. Review of Related Work

The nature of cyber-attacks against information and telecommunication systems is different and varies [7]. The information and telecommunication networks are protected from cyber-attacks using various tools and methods. All these tools and methods can be grouped into three groups: intrusion detection system (IDS), intrusion prevention system (IPS), intrusion response system (IRS).

One of the desirable features of IDS is being a real-time system. An adaptive intrusion detection system that can detect unknown attacks in real-time network traffic is a major concern. Conventional adaptive intrusion detection systems are computationally expensive in terms of computer resources and time because these systems have to be retrained with known and unknown attacks. Rathore et al. [8] proposed a real-time intrusion detection system for ultra-high-speed big data environment using Hadoop implementation. The proposed system is based on four-layered IDS architecture that consists of the capturing layer, filtration and load balancing layer, processing or Hadoop layer, and the decision-making layer. Al-Yaseen et al. [9] suggested a method that is based on a multi-agent system to allow the intrusion detection system to adapt to unknown attacks in real-time. The detection model uses the multi-level hybrid support vector machines and extreme learning techniques. Despite the widespread use of IDS systems, they have several weaknesses. Major deficiencies in the network intrusion detection systems (NIDS) include the inability to analyze encrypted traffic, late updates, time delay between attack start and warning, and the difficulty of processing data on a redundant network. Hybrid intrusion detection systems (HIDS) deficiencies are identified as failure to recognize network scans, inefficiencies in DoS attacks [10], [11]. Some IDSs can be relatively easily avoided (e.g., anomaly-based or signature based) [12], [13]. Werlinger et al. [14] state that the result of using IDS is not always clear. It is also interesting that practically the same imperfections have existed for many years [3], [15], [16] and even the new methods [13] [17] do not help to avoid them.

An IPS is a newer approach than the IDS to fight against the cyber security threat. The IPS combines the technique of firewall with the IDS [18]. The use of traditional IPSs for information and telecommunication systems is problematic for several reasons [19]:

1. Latency: in-bound IPS requires inspection and blocking action on each network packet, which consumes cloud system resources and increases the detection latency;
2. Resource Consumption: running the intrusion detection and prevention systems (IDPS) services usually consumes significant resources;
3. Inflexible Network Reconfigurations: traditional IPS does not have network configuration features to reconfigure the virtual networking system and provide scrutinized traffic inspection and control.

The IRSs are used for responding to attackers' actions. There are two types of an IRS: passive and active IRS, depending on the type of response. If a system automatically takes measures leading to a response, system is called an active IRS, if it takes place in a notification or forms a response in a manual way, system is called a passive IRS [20, 21]. The Audit expert system is currently widely used [21]. In support of Audit expert systems, Moon et al. [22] presented Multi-Layer Defense System (MLDS) that applies a reinforced defense system by collecting and analyzing log information and various information from network infrastructure. Heo et al. [23] suggested a system design that helps to maintain a certain level of quality of service and quality of security service in threatening environments. Nevertheless, despite all the advantages provided by such systems, they still have many deficiencies that are fully disclosed in the papers [16–21].

One of the biggest deficiencies is that such systems are susceptible to violations because they are relatively static (especially for the associative-based IRS). Other major deficiencies are the activation of such systems only when an incident is detected [20] and a high number of false alarms, which directly depends on the quality of IDS [21]. There are more deficiencies however they are related not to attack but to the healthy state of the system, which can be affected by the use or non-use of the IRS [20] or the use of appropriate hardware.

There are currently some attempts to detect cyber-attacks in the early stages. Yadav and Rao [3] suggested that the early stages include reconnaissance, weaponization, delivery, and initial part of the exploitation, in which, if an attack is detected, its effects can be eliminated. Siddique et al. [6] presented promising experimental results of the attack detection using IDS. Yan and Zhang [24] offered structured intrusion detection based on the behavioral semantics. However, it is not entirely clear how the early stage is understood and what opportunities are to process large flows of information. Vincent et al. [25] highlighted the importance of early detection and offered some solutions for detecting Trojan viruses. However, it should be noted that Vincent et al. [25] did not provide a detection algorithm. Chen et al. [26] proposed a model that integrated and correlated multiple logs to identify the early phase of targeted attacks. State-based hidden Markov model is used to detect joint attacks. However, this model is based on the IDS system, which, as noted above, has several shortcomings and, it is mostly designed to detect distributed denial-of-service (DDoS) attacks. Moreover, the idea of provided attack detection is vaguely presented. There are more investigations that are dedicated to the specific type of the attacks [27], [28]. Bhattacharya and Selvakumar [27] suggested a multi-measure multi-weight ranking approach for the identification of the network features for the detection of denial of service (DoS) and probe attacks. The approach combines the filter and wrapper feature selection methods and clustering methods to assign multiple weights to each feature. Cheng et al. [28] proposed a DDoS detection method for socially aware networking based on the time-series autoregressive integrated moving average model. The model describes a multi-protocol-fusion feature to characterize normal network flows.

The review shows that the tools and methods currently in place do not allow the effective control of threats in cyberspace. One of the reasons for such an ineffective fight is the fact that usually systems (IDS, IPS, and IRS) begin functioning only when the attack is already happening or even happened. Further reasons for relatively ineffective protection systems are the delay of the software updates and the ability to bypass or negatively impact protection systems functionality by exploiting their own vulnerabilities.

3. Early Stages of the Cyber-Attack

As already mentioned, the most dangerous cyber-attacks are those that are planned. The preparation of such attacks and their initial stages can last quite long – for months or even years. The attacker can assess all the victim's weaknesses and the consequences of the attack would be extremely harmful. The main purpose of these attackers is to get the user's access to the system, so their attack vectors are directed to obtain user rights in the system, exploiting system software vulnerabilities. At their late stages, such attacks normally cannot be terminated without causing losses. Such prepared cyber-attacks are difficult to detect because of their well-planned steps, but if they occur and enter the late stage sector, their consequences are the greatest comparing to other types of attacks. It would be advisable to distinguish two types of attacks: a classic attack and an intelligent attack.

The first type of attack is characterized by the fact that it has practically no individual stages, or in some cases, it is possible to distinguish one or two stages: exploration and attack. Such attacks are relatively fast, often without a well-defined target, they are poorly organized and coordinated. The tools used in such attacks are for creating rugged effects, i.e. launching DoS and DDoS attacks, various viruses (untargeted), malware, and the similar ones. Such attacks cause losses, but usually these losses relate to a single entity or individual object, the effects of such attacks are relatively easy recoverable, and the attackers are easily detectable. Attackers creating such attacks are normally represented in relatively low-impact output groups, i.e. hackers, crackers, phreakers or vandals. Normally, these groups are formed from a small number of members and in the most cases, just one member forms a group.

The second type of attacks (intelligent attacks) has the following characteristics: detailed planning, many stages, and slow progress. Attackers have a well-defined target and a well-defined goal. Attacks use malware specifically designed for the target and deep self-disguise. The effects of such attacks are extremely damaging, requiring a lot of effort to eliminate the consequences of the attacks. Such attackers are in large groups and well-organized, with sufficient financial resources from criminal groups, terrorist organizations or state structures.

For intelligent attacks, it is necessary firstly define their possible stages. As stated in the introduction, the elaboration of those stages is an important factor in enabling the most accurate estimation of the initial stages of the attack, which have not yet done any harm, and which can still be described as "chain killing". Yadav and Rao [3] proposed a vector for intelligent attacks that is formed out of seven stages. Although Yadav and Rao [3] clearly distinguished early stages and late stages, our offer is to extend the number of groups into three:

1. Early stages;
2. Transitional stages;

3. Late stages.

The early stages include processes for data collection, target tracking and attack infrastructure. In the transitional stages, information from the early stages is used and actions are taken to weaken the victims' system (e.g., implementing a malicious code or process against the system, exploiting its vulnerabilities). This enables the access to the system. Thereafter, the late stage attack processes follow direct system take-over, specific data capture, or infrastructure removal procedures.

Table 1. General classification of cyber-attacks

Attacks parameters	Classic	Intelligent
Number of Stages	1–2	> 3
Speed of Attack	Fast	Slow
Attack types	DoS, DDoS, malware, virus	Classic and custom-made software tools purposefully delivered to a certain target and specifically adapted to victim's network and system configuration
Attacker types	Individual person or small groups	Criminal and terrorist groups, state structures
Target of an attack	Separate object or subject	Object groups, state institutions, economical branches, wide social groups
Attackers financial resources	Relatively small	Wide financial resources, in some cases, unlimited
Consequences	Relatively small, easily recovered	Hard losses, hardly recovered

Based on the literature analysis and our own experience, we suggest an extension of the number of stages proposed in [3] to nine by adding the stage of social engineering and incorporating the stage of evasion (Fig. 1).

In the social engineering phase (Stage 0), there is an attempt to extract certain information about future cyber-attack target (entity or object) with some information that would facilitate a cyber-attack using the psychological effects of human beings. Experts say that the impact of social engineering is almost impossible to avoid. Therefore, this is a good way to extract certain primary data. In this paper, this stage will not be discussed further because it is an information collection step that involves various social and psychological manipulation techniques. However, to the extent that it aims to obtain data for planning a cyber-attack, social engineering should be considered as the initial stage of a cyber-attack.

We suppose that an attack can be withheld if it was detected in the preliminary stages 1 – 3, i.e. reconnaissance, weaponization and delivery, since an attack detection during these stages allows killing or blocking the attack. Because of the continuing attack, noticeable damage starts directly interfering with system and network work. It is necessary to detect these processes until they reach the 4-th stage (stage 4 – exploitation).

The first stage of the attack (reconnaissance) consists of three actions: port scan, host scan and system version scan. Port scan is a scanning of the network ports using a SYN request. Host scan is a scanning of the nodes in the system and obtaining their IP addresses. Version scan is an obtaining of the version of the system.

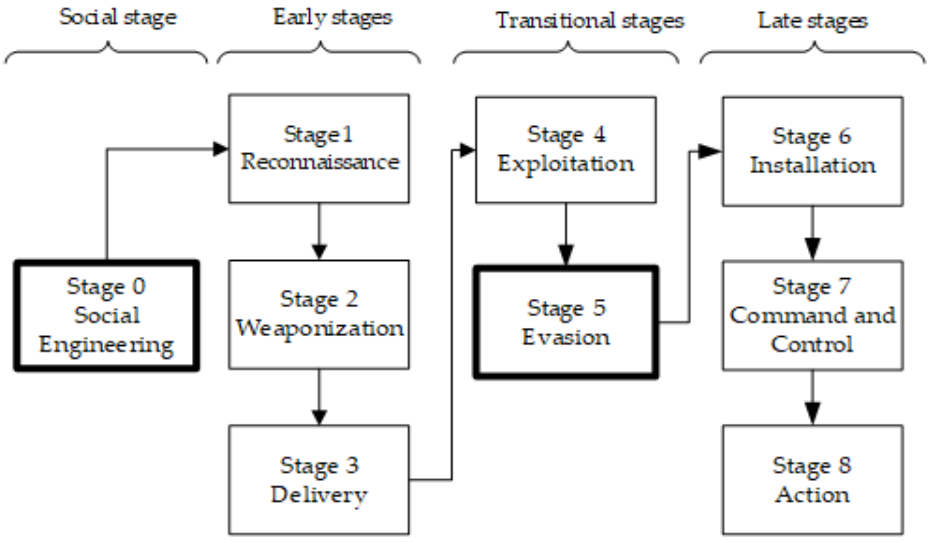


Fig. 1. Elaborated attack stages

The second stage (weaponization) includes two factors: system & services version scan and service stress test. The third stage (delivery) is a stage when the first part of the malicious code is delivered to victim's infrastructure to be executed at a certain time and it starts damaging processes against the targeted system. The third stage consists of two actions: version check and spoofing. Each of these actions has its own activities. For example, the spoofing action includes activities such as modifying network packets and programs with malicious code infiltration; performing stress tests over system processes remotely.

Processes and actions, which are executed by an adversary, can be registered by monitoring the network stack and system behavior. Results of the monitoring enable distinguishing the features inherent in these ongoing processes and application of them for detection of system anomalies and recognition of an attack to begin.

Attacks in the different stages have several characteristics that identify attack process. In this paper, we distinguish three characteristic groups that allow to characterize the ongoing processes: physical network stack parameters, logical parameters of the system being attacked, network stack flags.

4. Method to Detect the Early Stages of the Cyber-Attack

The determination provided in Section 3 of early stages of the cyber-attack enables the exploration of the ways to recognize the presence of such stages. The essence of the

proposed method is to use the appropriate logical filters to classify the certain parameters of the traffic. For this purpose, the total analysed data flow is considered to consist of two parts: the normal flow (i.e., the flow that is not harmful) and the attacker's flow (malicious flow). The generic filter consists of two blocks: a packet analysis block and a parameter processor. The traffic input into the filter is analyzed on the packet level, which results in a packet parameter (e.g., DST IP). The obtained parameter is passed to the internal parameter processor that forms an indicator value according to the conditions provided.

The detection method consists of three parts: filter part, evaluation block and action block (Fig. 2). The filters are implemented in two blocks: consolidated network filtering and system monitoring (CNFSM) and parameter preprocessing (PP). In the CNFSM block, the filters are grouped into three groups: filtering of network parameters (NF), filtering of system parameters (SF), filtering of network stack flags (LF). The evaluation block consists of three logical circuits that are connected at the outputs of the corresponding filter groups. The purpose of the filters is to register parameters and, if their values exceed predefined values, indicate the malicious activity. The purpose of the evaluation block is to collect the binary parameters and process them for the indication of the possible attack action. The purpose of the action block is to decide which stage of the attack is observed.

Using this principle, it is possible to analyze network traffic and system behavior adaptively by adjusting filters for analysis according to the need (available resources, depth of analysis, speed and tolerances of created system or network delays). To ensure early detection, different types of filters are used: network parameters NF (shown in circle); system parameters SF (depicted in rectangular); network stack flags LF (shown in hexagon). The three filter groups in total include 31 different filters: 12 filters belong to the NF group; 6 filters belong to the SF group and the remaining 13 filters belong to the LF group. These filters are consolidated, i.e. they perform the collection of the parameters and their analysis.

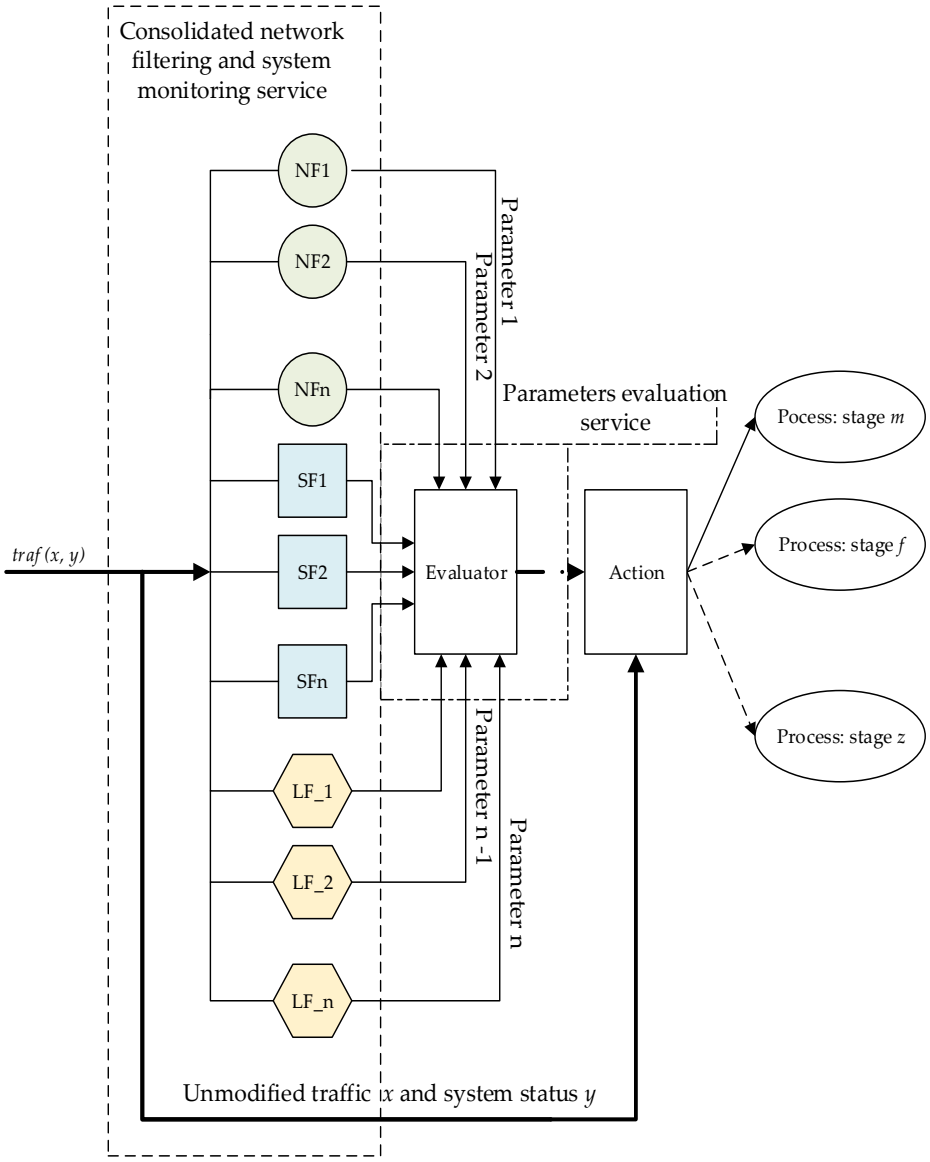


Fig. 2. Detailed schematic view of the filters

The filters of network parameters are numerated from 1 to 12 (NF1 ... NF12 and the outputs of the filters, which form inputs to the logical circuit, are labeled as x1 ... x12). The filters of system behavior are labeled from SF1 to SF6. The outputs of the filters are labeled as x13 ... x18. The filters of network stack flags are labeled from LF1 to LF13. The outputs of the filters are labeled as x19 ... x31. All the filters and their functions are enumerated in Table 2.

Table 2. Functions of attack monitoring filters

No.	Filter name	Filtering parameter	Filter description
1	NF1	IP	Attacker's IP address
2	NF2	IP COUNT	IP address repetition
3	NF3	PORT NUMBER	Port number to which the information is sent
4	NF4	PORT DISTRIBUTION	Distribution of ports according to the token information
5	NF5	PACKET COUNT	The number of packets in the network tract
6	NF6	STACK BYTES	Amount of data transferred in the session
7	NF7	PACKETS A-> B	Number of packets sent from the attacker to the victim
8	NF8	PACKETS B->A	Number of packets sent from the victim to the attacker
9	NF9	BYTES A->B	Amount of data transferred from the attacker to the victim
10	NF10	BYTES B->A	Amount of data transmitted from the victim to the attacker
11	NF11	DURATION	Duration of the active single session between the attacker and the victim
12	NF12	ABSOLUTE TIME	Absolute start time for the session
13	SF1	PERIPHERAL STATUS	Whether the peripheral device has changed
14	SF2	UNLISTED PROCESS	What processes in the system are in the list
15	SF3	FLAWLESS USER LOGIN	Whether an unexpected user connection was attempted or a password or unconnected

			connection was attempted
16	SF4	SUSPICIUOS TIME	System clock times which average is significantly deviating from standard user connection time
17	SF5	DISK ACTIVITY	Is the increased activity of the disk array detected by comparing with an average value
18	SF6	PORT BINDING	Whether the port is bound to port
19	LF_FIN	FIN FLAG	Packet's FIN flag
20	LF_SYN	SYN FLAG	Packet's SYN flag
21	LF_TCP_CONN()	TCP_CONN() FLAG	TCP Connection request
22	LF_NULL	NULL FLAG	NULL flag
23	LF_PING	ICMP FLAG	ICMP request
24	LF_VERSION_DETECTION	VER FLAG	VERSION flag
25	LF_UDP_SCAN	UDP FLAG	UDP request
26	LF_BULK_SCAN	BULK FLAG	Random request
27	LF_WINDOWS_SCAN	WIN_SCAN FLAG	Versions of Windows query
28	LF_RPC_SCAN	RPC FLAG	Identify the RPC protocol
29	LF_LIST_SCAN	LST FLAG	A query that results a list of the previous query vector
30	LF_IDLE_SCAN	IDL FLAG	An IDLE process request
31	LF_FTP_BOUNCE	BOUNCE FLAG	FTP service request

Data collected from all types of the logical filters is sent to the parameter preprocessing block, in which, according to filtered parameters, sets of attack parameters are further processed. If the value of the filtered parameter exceeds the predefined value, then this parameter is assigned a binary value 1 (anomaly value), otherwise – the binary value 0 (normal value). According to the result of this process, we can distinguish seven attack actions that fall into early three attack stages. The actions are as follows:

1. HS – Host Scan;
2. PS – Port Scan;
3. SSV – System and Services Version;
4. SST – Services Stress Tests;
5. SP – Spoofing;

6. LA – Login Attempt;
7. SE – Service Exploitation.

Processed session parameter sets are sent to the evaluation block of logical circuits. The evaluation block consists of three independent logic circuits: the first logical circuit performs analysis of evaluated parameters of NF, the second logical circuit evaluates the analyzed SF parameters, the third logical circuit performs analysis of filtered LF parameters. The configuration of these filters allows to create a setup of the detection, the result of which is determined by the logical circuits.

The logical circuits operate on the sets of binary parameters. Seven types of the possible attack actions were determined; therefore, we have designed the logical circuits having seven primary outputs. In such a way, every primary output indicates the presence of the different attack action. A logical circuit of NF analysis uses primary inputs x1 ... x12 and produces seven primary outputs labeled as F21 ... F27. A logical circuit of SF analysis uses primary inputs x13 ... x18 and produces seven primary outputs labeled as F35 ... F41. A logical circuit of LF analysis uses primary inputs x19 ... x31 and produces seven primary outputs labeled as F42 ... F48. Subsequently, the primary outputs of all the logical circuits are combined at the final point in the evaluator block.

The bit stream, which arrives at the inputs of the evaluation block, is divided into three parts and supplied into three independent logical circuits. Every circuit is dedicated and produces seven bits. The values at the primary outputs of all three logical circuits are joint into single vector. Only the values of the combined vector can implicate the presence of the attack action. The presence of the values of the final vector in the lookup table indicates the early stage of the cyber-attack.

The analytical form of logical circuit of NF analysis is shown in (1). A member xA, where $A \in \{1 \ ... \ 12\}$, corresponds to the binary 1, and a member $\overline{xA}$, where $A \in \{1 \ ... \ 12\}$, corresponds to the binary 0. This form contains output logical functions, which consist of inputs $x1$ ... $x12$ and Output 1 is a vector of F21 ... F27 values.

$$OUTPUT1 = \begin{Bmatrix} F21 \\ F22 \\ F23 \\ F24 \\ F25 \\ F26 \\ F27 \end{Bmatrix} = \cdot \begin{Bmatrix} x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot \overline{x5} \cdot \overline{x6} \cdot \overline{x7} \cdot \overline{x8} \cdot \overline{x9} \cdot \overline{x10} \cdot \overline{x11} \cdot \overline{x12} \\ x1 \cdot x2 \cdot x3 \cdot x4 \cdot \overline{x5} \cdot \overline{x6} \cdot \overline{x7} \cdot \overline{x8} \cdot x9 \cdot x10 \cdot \overline{x11} \cdot \overline{x12} \\ x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot x5 \cdot \overline{x6} \cdot \overline{x7} \cdot \overline{x8} \cdot \overline{x9} \cdot \overline{x10} \cdot \overline{x11} \cdot \overline{x12} \\ x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot \overline{x5} \cdot x6 \cdot x7 \cdot x8 \cdot \overline{x9} \cdot \overline{x10} \cdot \overline{x11} \cdot \overline{x12} \\ x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot x5 \cdot \overline{x6} \cdot \overline{x7} \cdot \overline{x8} \cdot \overline{x9} \cdot \overline{x10} \cdot x11 \cdot x12 \\ x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot x5 \cdot \overline{x6} \cdot \overline{x7} \cdot \overline{x8} \cdot \overline{x9} \cdot \overline{x10} \cdot x11 \cdot x12 \\ x1 \cdot x2 \cdot \overline{x3} \cdot \overline{x4} \cdot x5 \cdot x6 \cdot \overline{x7} \cdot \overline{x8} \cdot \overline{x9} \cdot \overline{x10} \cdot \overline{x11} \cdot \overline{x12} \end{Bmatrix} \quad (1)$$

Table 3 shows the attack actions that make up the attack vector. The values in the column under name “Action” have the attribute “part” because the single circuit on its own cannot define fully the action of the attack. The action of the attack can be defined only when the results of the all three circuits are combined.

Table 3. Lookup table of *x*1…*x*12 parameter set and NF output values

No.	ACTION	INPUTS												OUTPUTS						
		x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	F21	F22	F23	F24	F25	F26	F27
1	HS part	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
2	PS part	1	1	1	1	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0
3	SSV part	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
4	STT part	1	1	0	0	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0
5	SP part	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0
6	LA part	1	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0
7	SE part	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1

The logical circuit of NF analysis is presented in Fig. 3. On the left side of the picture, it is marked binary processed NF parameter inputs, these parameters are obtained directly from the network driver.

The logical circuit uses 23 logical gates. The primary outputs F25 and F26 are identical due to the identity of the parameter values analyzed (the parameters of the SP Part and LA part analyzed are identical in this analysis, so the generated response is the same, but the outputs are different).

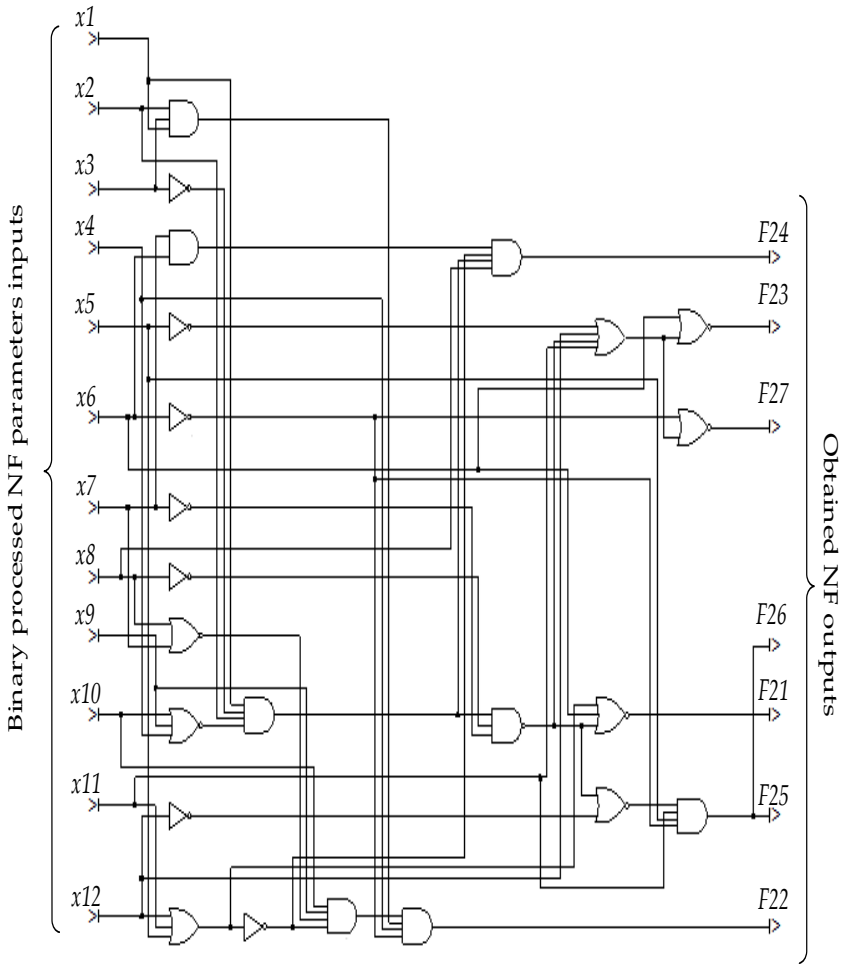


Fig. 3. Logic circuit of x1…x12 bit stream parameters

For example, the values on the primary inputs indicating the HS action part of the attack are as follows: $x1 = 1$, $x2 = 1$, $x3 = 0$, $x4 = 0$, $x5$ ... $x12 = 0$, and the primary output F21 = 1, the remaining primary outputs F22 ... F27 = 0. In this case, the HS action part of the attack will be detected when the parameter $x1$ "IP address" and the parameter $x2$ "IP repetition" exceed their predefined values, meanwhile, the predefined values of the remaining NF filter parameters $x3$ ... $x12$ will not be exceeded. In this case, the entire output vector will have a value of 1000000. Such an assessment is only part of the overall assessment of the HS action process, and other parts of the assessment are performed at SF and LF logical circuits, respectively.Output 1 is the first part of the logical analysis results, further results are obtained from SF and LF analysis, named as Output 2 (SF) and Output 3 (LF), respectively.

Table 4. Lookup table of x13…x18 parameter and SF output

No.	ACTION	INPUTS						OUTPUTS						
		x13	x14	x15	x16	x17	x18	F35	F36	F37	F38	F39	F40	F41
1	HS part	0	0	0	0	0	0	1	0	0	0	0	0	0
2	PS part	0	0	0	0	0	0	0	1	0	0	0	0	0
3	SSV part	0	0	0	0	0	1	0	0	1	0	0	0	0
4	STT part	0	1	1	1	1	1	0	0	0	1	0	0	0
5	SP part	0	0	0	0	0	1	0	0	0	0	1	0	0
6	LA part	0	0	1	1	0	1	0	0	0	0	0	1	0
7	SE part	0	1	1	1	1	1	0	0	0	0	0	0	1

$$OUTPUT\,2 = \begin{Bmatrix} F35 \\ F36 \\ F37 \\ F38 \\ F39 \\ F40 \\ F41 \end{Bmatrix} = \begin{Bmatrix} \overline{x13} \cdot \overline{x14} \cdot \overline{x15} \cdot \overline{x16} \cdot \overline{x17} \cdot \overline{x18} \\ \overline{x13} \cdot \overline{x14} \cdot \overline{x15} \cdot \overline{x16} \cdot \overline{x17} \cdot \overline{x18} \\ \overline{x13} \cdot \overline{x14} \cdot \overline{x15} \cdot \overline{x16} \cdot \overline{x17} \cdot x18 \\ \overline{x13} \cdot x14 \cdot x15 \cdot x16 \cdot x17 \cdot x18 \\ \overline{x13} \cdot \overline{x14} \cdot \overline{x15} \cdot \overline{x16} \cdot \overline{x17} \cdot x18 \\ \overline{x13} \cdot \overline{x14} \cdot x15 \cdot x16 \cdot \overline{x17} \cdot x18 \\ \overline{x13} \cdot x14 \cdot x15 \cdot x16 \cdot x17 \cdot x18 \end{Bmatrix} \quad (2)$$

Table 4 shows the attack actions that make up the attack vector of the SF analysis. In this case, the attack action HS part and the attack action PS part describe the values on the primary inputs *x*13 ... *x*18 as zeroes. This is because the SF analysis parameters *x*13 ... *x*18 do not take part in forming HS and PS actions. However, this result is important, the outputs F35 and F36 are assigned the appropriate values. The logical circuit used for SF analysis is shown in Fig. 4. The analysis is based on six criteria, so there are six primary inputs and, as previously mentioned, seven primary outputs to identify action of the attack.

In the logical circuit of SF analysis, 12 logical gates are used. As in the case of NF circuit, there are input sequences that are identical, therefore, the primary outputs F35 and F36, the primary outputs F37 and F39, the primary outputs F38 and F41 are connected in parallel. Even though some input vectors for the SF circuit are the same, their combination with input vectors of the NF circuit makes the unique input vector and produces a different final output result.

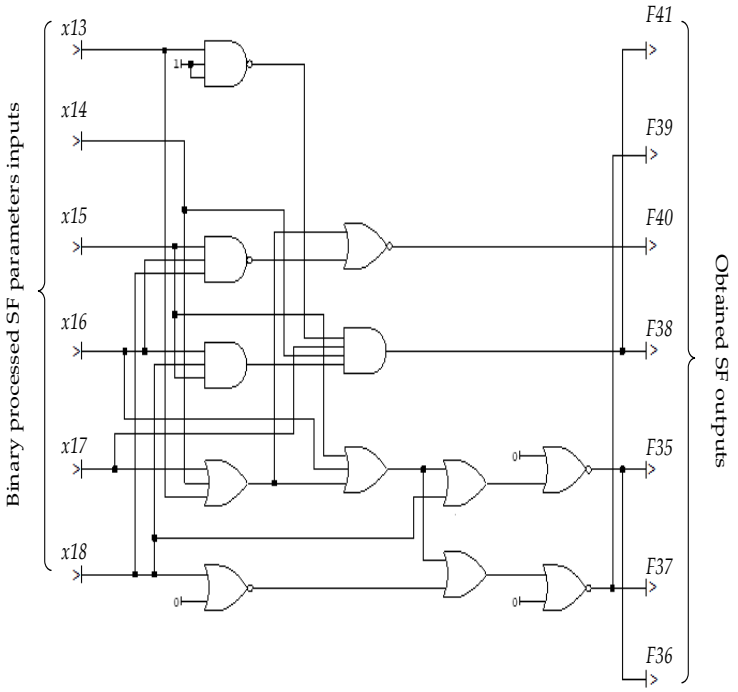


Fig. 4. Logic circuit of x13…x18 bit stream parameters

Analytical approach to the LF Analysis Output 3 result is shown in the (3). Analogically to the Output 1 and Output 2 forms, a member xA, $A \in \{19 \ldots 31\}$, corresponds to the binary 1, and a member $\overline{xA}$, $A \in \{19 \ldots 31\}$, corresponds to the binary 0. Output 3 is the last component of the logical gate analysis, characterizing the flag states in the network stack. The values on the primary outputs of the logical circuits are combined and the attack factors are determined according to the obtained result.

Table 5 shows the attack actions that make up the attack vector of the LF analysis. In this case, the primary inputs x19 ... x31 and the primary outputs F42 ... F48 are used. Differently from NF and SF analysis, there are no duplicate output cases. All the primary outputs are activated with unique combinations on the primary inputs

$$OUTPUT3 = \begin{Bmatrix} F42 \\ F43 \\ F44 \\ F45 \\ F46 \\ F47 \\ F48 \end{Bmatrix} = \begin{Bmatrix} \overline{x19}\cdot\overline{x20}\cdot\overline{x21}\cdot\overline{x22}\cdot x23\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \\ x19\cdot x20\cdot\overline{x21}\cdot\overline{x22}\cdot\overline{x23}\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \\ \overline{x19}\cdot\overline{x20}\cdot x21\cdot x22\cdot\overline{x23}\cdot x24\cdot x25\cdot x26\cdot x27 \\ \overline{x19}\cdot\overline{x20}\cdot x21\cdot\overline{x22}\cdot x23\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \\ \overline{x19}\cdot\overline{x20}\cdot x21\cdot x22\cdot\overline{x23}\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \\ \overline{x19}\cdot\overline{x20}\cdot x21\cdot\overline{x22}\cdot x23\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \\ \overline{x19}\cdot\overline{x20}\cdot x21\cdot x22\cdot x23\cdot\overline{x24}\cdot\overline{x25}\cdot\overline{x26}\cdot\overline{x27} \end{Bmatrix} \cdot \begin{Bmatrix} \overline{x28}\cdot\overline{x29}\cdot x30\cdot\overline{x31} \\ \overline{x28}\cdot\overline{x29}\cdot\overline{x30}\cdot\overline{x31} \\ x28\cdot x29\cdot\overline{x30}\cdot\overline{x31} \\ \overline{x28}\cdot\overline{x29}\cdot\overline{x30}\cdot x31 \\ \overline{x28}\cdot\overline{x29}\cdot\overline{x30}\cdot\overline{x31} \\ \overline{x28}\cdot\overline{x29}\cdot\overline{x30}\cdot\overline{x31} \\ \overline{x28}\cdot\overline{x29}\cdot x30\cdot x31 \end{Bmatrix}.$$

Table 5. Lookup table of *x*18…*x*31 parameter set and LF output values

No.	ACTIONS	INPUTS													OUTPUTSS					
		x19	x20	x21	x22	x23	x24	x25	x26	x27	x28	x29	x30	x31	F42	F43	F44	F45	F46	F47	F48
1	HS part	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
2	PS part	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
3	SSV part	**0**	0	1	1	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
4	STT part	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
5	SP part	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6	LA part	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
7	SE part	0	0	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1

The logical circuit used for LF analysis is shown in Fig. 5. For the analysis, 13 criteria are used and 13 primary inputs x19 ... x31 correspond to them. The seven primary outputs F42 ... F48 for identifying attack actions are used. The logical circuit consists of 26 logical gates. In this circuit, unlike NF and SF cases, there are no identical value combinations on the primary inputs.

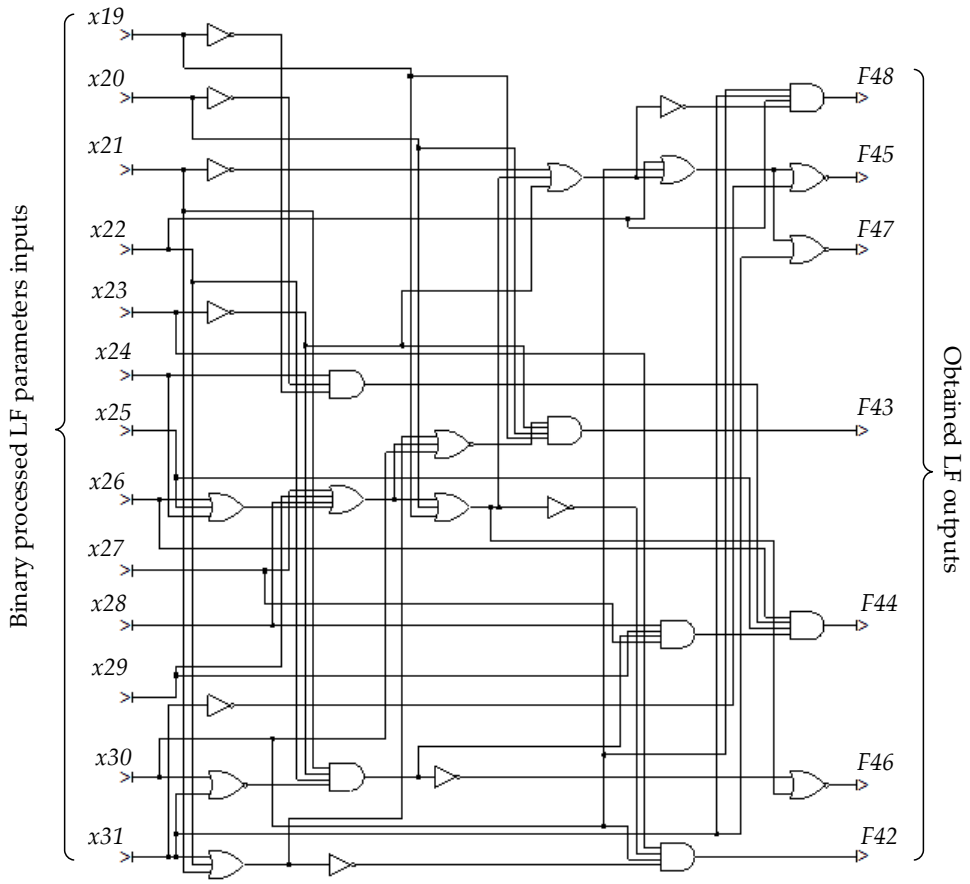


Figure 5. Logic circuit of *x19*…x31 bit stream parameters

The analytical aggregated expression combining the previously presented separate results is presented in (4). Formula (4) combines output vectors of Output 1, Output 2, and Output 3 into single vector for the cyber-attack detection. For example, HS denotes that attack action Host Scan is fully characterized by the code HS = 100000010000001000000 consisting of a set of NF = {F21 ... 27}, SF = {F35 ... 41} and LF = {F42 ... F48} filters. We can determine the early stage of the attack according to the values in Table 6.

$$OUTPUT\ ACTION\ FULL\ (OAF) = OUTPUT1 + OUTPUT2 + OUTPUT3 \quad (4)$$

Table 6. Lookup table for definition of attack actions

No.	ACTIONS	NF OUTPUTS							SF OUTPUTS							LF OUTPUTS						
		F21	F22	F23	F24	F25	F26	F27	F35	F36	F37	F38	F39	F40	F41	F42	F43	F44	F45	F46	F47	F48
1	HS	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0
2	PS	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
3	SSV	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
4	STT	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
5	SP	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
6	LA	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
7	SE	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1

5. Experiment

We carried out the experiments on the synthesized data to evaluate the capabilities of the proposed method. The experimental set consists of two parts:

1. Attack vector: values from the binary set {0, 1} are filled in deterministically using the determined attack parameters.
2. Random vector sequence: values from the binary set {0, 1} are generated randomly.

To determine the detection capabilities of the proposed method, we have generated an array of 100352 events that were analysed by the proposed logical circuits. As described in Section 4, the proposed early detection method consists of three filters: network filter (NF), system filter (SF) and network flags filter (LF). The generated array was analysed

by these filters separately and the obtained results were combined to determine the attack action formed out of events. The generated 100352 events consist of 100053 randomly generated events and 299 events that have the parameters of known attacks. NF, SF and LF filters identification values are shown in Table 7. Eight parameters are shown in the table of the filters: seven parameters that indicate a detection of an attack: HS, LA, PS, SE, SP, SST, SSV, and a parameter 0 that corresponds to non-malicious traffic.

Table 7. NF, SF and LF filters attack identification value

Actions	Filter NF			Filter SF			Filter LF		
	No. of events	Randomly generated events	Deterministically generated events	No. of events	Randomly generated events	Deterministically generated events	No. of events	Randomly generated events	Deterministically generated events
HS	6242	95%	5%	7632	96%	4%	3728	92%	8%
LA	1790	83%	17%	2133	86%	14%	4068	93%	7%
PS	438	32%	68%	7632	96%	4%	2979	90%	10%
SE	944	68%	32%	888	66%	34%	6251	52%	48%
SP	1790	83%	17%	6997	96%	4%	3535	92%	8%
SST	616	51%	49%	888	66%	34%	2162	86%	14%
SSV	1802	83%	17%	6997	96%	4%	3429	14%	86%
0	8610	10%	0%	6710	10%	0%	8210	10%	0%

As it was possible to predict, the largest values are for non-malicious traffic, which are shown in the last row of Table 7. The values 0% in this row show the very important obtained result that all the deterministically generated events were detected as malicious. The biggest number of detected events in the filter NF was HS and the lowest – PS. HS actions also had a high detection ratio in the filter SF. The mostly noticeable difference between the filters SF and NF was that PS action in the filter SF had a high detection ratio of randomly generated events. The filter LF showed much smaller number of events in comparison with the filters NF and SF for action HS. Such differences in action detection ratios among filters show methods specificity. In the Table 8, an aggregated form out of three filters for actions HS, LA, and PS is shown, which indicates the detection of the first stage – Reconnaissance. The obtained result confirms that the proposed method is able to

detect all the deterministically on purpose generated events. The proposed method has also detected a number of randomly generated events, which varies depending on the action.

For the second part of the experimental set, we have generated an array of randomly selected 100352 events. The objective of this part of the experiment is to evaluate the possibility to create an attack randomly. The results of three filters NF, SF and LF and accumulation results $A = NF$ & SF& LF are shown in Table 9.

Table 8. Aggregated form of three filters

Name	HS	LA	PS
Detected events	1154	305	316
Detected randomly generated events	855	6	17
Detected deterministically generated events	299	299	299
Deterministically/TOTAL %	26%	98%	95%

Table 9. The results of filters NF, SF, LF and accumulated detection

Action	Filter *NF*		Filter *SF*		Filter *LF*		Accumulated detection	
	No of detected events	% of total events	No of detected events	% of total events	No of detected events	% of total events	No of detected events	% of total events
HSL	6424	6%	7616	8%	3790	4%	1181	1%
A	1784	2%	2165	2%	4107	4%	301	0,3%
PS	466	0%	7616	8%	2922	3%	312	0,3%
SE	967	1%	874	1%	609	1%	289	0,3%
SP	1784	2%	6653	7%	3495	3%	310	0,3%
SST	621	1%	874	1%	2148	2%	270	0,3%
SSV	1803	2%	6653	7%	320	0%	2	0%
"0"	86503	86%	67901	68%	82961	83 %	97687	97,6 %

The main part of the traffic is generated randomly for the both parts of the experiment. Therefore, we can compare a detection of malicious actions in random traffic in Table 7 and in Table 9. The main stream of the randomly generated traffic is non-malicious (see the last rows of Table 7 and Table 9). Moreover, the obtained numbers of the non-malicious traffic are quite similar in both tables, e.g. filter NF showed 86730 events for action HS in Table 7 and filter NF showed 86503 events for action HS in Table 9. The same is true for detection of malicious events, as well. For example, filter SF showed 2133 events for action LA in Table 7 and filter SF showed 2165 events for action LA in Table 9.

We carried out the experiment on the real attack data that were taken from open databases. The experiment was carried out on the network of virtual machines. The results of the experiment are presented in Table 10.

Table 10. The results of detection of real attacks

No.	Simple or complex attack	Type of attack	Number of attacks	Detection (%)
1.	Simple targeted attack	Syn Flood	37	92 %
		Ack Flood	22	91 %
		IP fragment attack	20	80 %
		Xmas scan	16	81 %
		Password Bruteforce	70	87 %
2.	Complex attack	Cryptolocker	20	90 %
		Wannacry	15	80 %

We can observe (see Table 10) that our proposed method can detect the real simple and complex cyber-attacks at their early stages. Not all the cyber-attacks are detected. The least percent of the detection is 80. Not all the cyber-attacks follow our introduced rules for the attack detection at the early stages.

Our experiments confirmed that the proposed method is capable to detect the early stages of the cyber-attacks in the network traffic. The method showed that the randomly generated traffic consists of 1,6% events indicating reconnaissance stage (the first stage), 0,3% events indicating weaponization stage (the second stage) and 0,3 % events indicating delivery stage (the third stage). The proposed method is a part of a larger work that is oriented to a near real-time cyber-attack detection.

6. Conclusions

Scientific and technical literature analysis and good practice show that the current system of response to cyber threats using IDS, IPS and IRS systems has a number of shortcomings, the main problem is that they start up only when a cyber-attack is taking place, i.e. such a system does not play a preventive role.

Intelligent cyber-attacks are characterized using certain stages. To determine the precautionary stage, when preventive measures can "kill the chain", the identification of those stages must be complete as possible. In this paper, we suggested to consider an attack chain of nine steps to describe a cyber-attack vector.

The paper proposes a method to detect an intelligent cyber-attack, which takes several preparation steps, and which is the most dangerous one, in the early stages of the cyber-attack. The method is based on the use of several logical filters. We have built the analytical aggregated expressions for the detection of threats caused by the early stages of the cyber-attacks.

The mode to detect the early stages of the cyber-attack may be appropriate for both standard information systems and small-sized mobile devices, since the suggested method is suitable for processing data on devices with a limited memory and computing power.

The experiments to test the ideas implemented in the proposed method were carried out. The essence of the experiments was to evaluate the reliability of the suggested method. All the values, which were generated deterministically for the attack, were identified as the malicious ones. The proposed method was able to detect many real simple and complex cyber-attacks at their early stages. In our opinion, such a result shows a good base for further work in increasing the sensitivity of the method to other forms of the cyber-attacks.

References

1. Symantec.: Preparing of a cyber-attack. Available online: http://symc.ly/1PHHl3n, accessed on 02 June 2018.
2. Kearney, A. T.: Information security: preparing for the next hack attack. Available online: http://bit.ly/2vtIwkM, accessed on 30 June 2018.
3. Yadav, T., Rao, A. M.: Technical aspects of cyber kill chain. Proc. of Security in Computing and Communications: Third International Symposium on Security in Computing and Communication, Kochi, India, 10-13 August 2015, pp. 438-452, Springer, Switzerland.
4. Husak, M.: Early detection and mitigation of multi-stage network attacks. PhD thesis, Masarykova Univerzita Fakulta Informatiky, Brno, Czech. Available online: https://is.muni.cz/th/ccz8a/thesis_proposal.pdf, accessed on 12 July 2018.
5. Morinaga, M., Nomura, Y., Furukawa, K., Temma, S.: Cyber-attack countermeasure technologies using analysis of communication and logs in internal network. Fujitsu Scientific and Technical Journal, 52 (3), 2016, 66-71.
6. Siddique, K., Akhtar Z., Lee H., Kim, W., Kim, Y.: Toward bulk synchronous parallel-based machine learning techniques for anomaly detection in high-speed big data networks. Symmetry, 9(9), 2017, 197.
7. Sharifi, A. A., Noorollahi, B. A., Farokhmanesh, F.: Intrusion detection and prevention systems (IDPS) and security issues. International Journal of Computer Science and Network Security. 14 (11),2014, 80-84.
8. Rathore, M.M., Ahmad, A., Paul, A.: Real time intrusion detection system for ultra-high-speed big data environments. J Supercomput, 72, 2016, 3489-3510.
9. Al-Yaseen W. L., Othman A. L., Nazri M. Z. A.: Real-time multi-agent system for an adaptive intrusion detection system. Pattern Recognition Letters, 85,2017, 56–64.
10. SANS Institute InfoSec Reading Room.: IDS - today and tomorrow. Available online: https://www.sans.org/reading-room/whitepapers/detection/ids-today-tomorrow-351, accessed on 22 August 2018.
11. Rajan, S. S., Cherukuri, V. K.: An overview of intrusion detection systems. Proc. IDT Workshop on Interesting Results in Computer Science and Engineering (IRCSE). Available online:https://pdfs.semanticscholar.org/012c/3afab09d34bad3d62cb499f2f57c40675062.pdf, accessed on 22 August 2018.
12. Guevara, C., Santos, M., López V.: Data leakage detection algorithm based on task sequences and probabilities. Knowledge-Based Systems, 120, 2017, 236-246.

13. Kashyap, S., Agrawal, P., Pandey, V. C., Keshri, S. P.: Importance of intrusion detection system with its different approaches. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, 2 (5), 2013, 1902-1908.
14. Werlinger, R., Hawkey, K., Muldner, K., Jaferian, P., Beznosov, K.: The challenges of using an intrusion detection system: is it worth the effort? Proc. of ACM Symposium on Usable Privacy and Security (SOUPS), Pittsburgh, Pennsylvania, USA, 23-25 July,2008, pp. 107-118, ACM, New York.
15. Lo, C.H., Ansari, N.: Consumer: a novel hybrid intrusion detection system for distribution networks in smart grid. IEEE Transactions on Emerging Topics in Computing, 1 (1),2013, 33-44.
16. Singh, K., Tamrakar, S.: A review of intrusion-detection system- clustering and classification using RBF and SOM networks. International Journal of Emerging Technology and Advanced Engineering, 5 (7), 2015, 502-505.
17. Ghazi Z., Doustmohammadi A.: Intrusion detection cyber-physical systems based on Petri net. Information Technology and Control, 47 (2), 2018, 220-235.
18. Chi, Y., Jiang, T., Li, X., Gao C.: Design and implementation of cloud platform intrusion prevention system based on SDN. Proc. 2nd IEEE International Conference on Big Data Analysis (ICBDA), Beijing, Peoples R China, March 10-12, 2017, pp. 847-852.
19. Xing, T., Huang, D., Xiong, Z., Medhi, D.: SDNIPS: enabling software-defined networking-based intrusion prevention system in clouds. Proc. of International Conference on Network and Service Management (CNSM), Rio de Janeiro, Brazil, 17-21 November,2014, pp. 308-311.
20. Ragsdale, D. J., Carver, C. A., Humphries, J. W., Pooch, U. W.: Adaptation techniques for intrusion detection and intrusion response systems. Proc. of the IEEE International Conference on Systems, Man, and Cybernetics, Nashville, TN, USA, 8-11 October,2000, pp. 2344–2349.
21. Shameli-Sendi, A., Cheriet, M., Hamou-Lhadj, A.: Taxonomy of intrusion risk assessment and response system. Computers and Security. 45, 2014, 1–16.
22. Moon, D., Im, H., Lee, J. D., Park, J. H.: MLDS: Multi-layer defense system for preventing advanced persistent threats. Symmetry. 6, 2014, 997-1010.
23. Heo, S., Lee, S., Doo, S., Yoon, H.: Design of a secure system considering quality of service. Symmetry, 6,2014 938-953.
24. Yan, X., Zhang, J. Y.: Early detection of cyber security threats using structured behavior modeling. ACM Transactions on Information and System Security. Available online:http://www.cs.cmu.edu/~xiaohuay/papers/draft_TISSEC.pdf, accessed on 12 July 2018.
25. Vincent, H., Wells, L., Tarazaga, P., Camelio, J.: Trojan detection and side-channel analyses for cyber-security in cyber-physical manufacturing systems. Procedia Manufacturing, 1, 2015,77-85.
26. Chen, M. C., Yang, P. Y., Ou, Y. H., Hsiao, H. W.:Targeted attack prevention at early stage. Proc. 28th International Conference on Advanced Information Networking and Applications Workshops. Victoria, BC, Canada, 13-16 May, pp. 866-870, 2014.
27. Bhattacharya, S., Selvakumar, S.: Multi-measure multi-weight ranking approach for the identification of the network features for the detection of DoS and probe attacks. *The Computer Journal*, 59 (6),2016, 923–943.
28. Cheng, J., Zhou, J., Liu, Q., Tang, X., Guo, Y.: A DDoS detection method for socially aware networking based on forecasting fusion feature sequence. The Computer Journal, 61 (7),2018, 959–970.

Vacius Jusas graduated from Kaunas Polytechnic Institute (Lithuania) in 1982. He received the D.Sc. degree from Kaunas Polytechnic Institute in 1988. Since 2006, he is professor at Department of Software Engineering, Kaunas University of Technology, Lithuania. He is author and co-author of more than 100 papers. He is Editor of journal "Information Technology and Control". His research interests include brain-computer interface, adaptive signal processing, forensics investigation, cybercrime.

Saulius Japertas graduated Vilnius University in 1982. He received his Ph.D. degree from the Lithuanian Energy Institute in 1991. Since 1992 he joined to Lithuania Army Air forces as a head of communication department and from 2003 to 2009 (retired) was a head of CIS Service under MoD. Since 1999 to 2018, he was an assoc. prof. in Electrical and Electronics faculty and since 2018 he is the assoc. prof in Mechanical engineering and design faculty, Kaunas University of Technology. He is author and co-author of more than 40 papers. His research interests include wireless networks, security and protection of electronics and IT&T networks.

Tautvydas Baksys graduated Kaunas University of Technology (Lithuania) in 2012. He received Master degree in Electronics. Now, he is a PhD student in Electronics and Electrical Engineering at Kaunas University of Technology. He is author and co-author of 3 papers. His research interests include wireless networks, security and protection of electronics and IT&T networks.

Sandeepak Bhandari graduated from I.K. Gujral Punjab Technical University (India) in 2013. He received Master of Technology in Computer Science and Engineering from I.K. Gujral Punjab Technical University. Now, he is a PhD student in Electronics and Informatics Engineering at Kaunas University of Technology. He is author and co-author of 14 papers. His research interests include wireless networks and security, forensics investigations and Data mining.

Received: January 22, 2019; Accepted: May 20, 2019

Computer Science and Information Systems 16(2):515–539 https://doi.org/10.2298/CSIS180312039B

Majority Vote Feature Selection Algorithm in Software Fault Prediction

Emin Borandag[1], Akin Ozcift[1], Deniz Kilinc[1], Fatih Yucalar[1]

[1] Department of Software Engineering, Faculty of Technology,
Manisa Celal Bayar University, Manisa, Turkey
{emin.borandag, akin.ozcift, deniz.kilinc, fatih.yucalar}@cbu.edu.tr

Abstract. Identification and location of defects in software projects is an important task to improve software quality and to reduce software test effort estimation cost. In software fault prediction domain, it is known that 20% of the modules will in general contain about 80% of the faults. In order to minimize cost and effort, it is considerably important to identify those most error prone modules precisely and correct them in time. Machine Learning (ML) algorithms are frequently used to locate error prone modules automatically. Furthermore, the performance of the algorithms is closely related to determine the most valuable software metrics. The aim of this research is to develop a Majority Vote based Feature Selection algorithm (MVFS) to identify the most valuable software metrics. The core idea of the method is to identify the most influential software metrics with the collaboration of various feature rankers. To test the efficiency of the proposed method, we used CM1, JM1, KC1, PC1, Eclipse Equinox, Eclipse JDT datasets and J48, NB, K-NN (IBk) ML algorithms. The experiments show that the proposed method is able to find out the most significant software metrics that enhances defect prediction performance.

Keywords: software fault prediction, majority voting, machine learning algorithm

1. Introduction

Prediction of software defects with the use of software fault prediction models is a cost-effective approach in the usage of limited project resources. Static software measures, i.e. size, coupling, cohesion, inheritance, complexity measures and defect data collected may be used to construct machine-learning methods to predict faults in practice. Quality of software modules are predicted as fault-prone (*fp*) and not-fault-prone (*nfp*). In this context, if an error is acquired during module tests, the corresponding module is marked as *fp* otherwise *nfp*. Defects in software projects lead to failures that increase the total cost of the project. From software developer point of view, usage of efficient fault investigation methods is important to predict defect-prone modules and thus improve software quality [1-3]. Software prediction models make use of historical software project faults and its corresponding software metrics to identify quality of an upcoming project/module from similar domain. Prediction performance of a software quality model depends on the information represented with software metrics. Therefore, software quality estimation models require selection of relevant metrics to improve their discrimination ability. Feature selection methods are used to obtain a subset of valuable

software attributes among all. In this context, development of a feature selection model is the main focus of this study [3].

Investigation of relevant metrics is a search problem and the answer to this problem is the exploration of software metric space with the use of feature selection strategies. The feature selection process attempt to locate the feature subsets that represent the data at least as good as the original data with all features. In particular, the feature selection strategies are classified in two groups, i.e. filtering and wrapper feature subset selection algorithms [4]. Filtering based algorithms makes use of some statistical criteria to arrange attributes according to their importance or weights. On the other hand, wrapper methods locate the most predictive feature subset with the use of search algorithms. It is expected that relevant feature subsets may produce a better prediction ability compared to the features alone [5]. In this study, we evaluate filtering based feature selection algorithms to obtain an effective feature subset.

The problem with subset selection is that evaluation of whole candidate metric subsets is ineffective in terms of computational resources. Therefore, we explain our Majority Vote based Feature Selection (MVFS) strategy in having two steps. First we rank the metrics according to their relative importance with the help of 4 well-known feature filtering strategies, i.e. Information Gain (IG), Symmetrical Uncertainty (SU), ReliefF (RLF) and Correlation-based (CO) [6], second, we select the relevant metric subset with a voting scheme borrowed from ensemble learning domain [7]. In this strategy, each feature in the subset is obtained with the majority votes of the feature filtering algorithms on the feature. Having obtained feature subsets with the proposed strategy, we make use of 3 machine learning algorithms i.e. Naïve Bayes (NB), Decision Tree (J48), and K Nearest Neighbor (K-NN/IBk) [8], to evaluate the defect prediction ability of corresponding software metrics. The experimental results show that gradual decrease of software metric space with the proposed MVFS algorithm increases performance of the models.

The main contribution of the study is following: Basic feature filtering strategies are better to be combined in some way to obtain an improved fault prediction performance. In the software fault prediction literature, there are many hybrid strategies that combine feature selection strategies to obtain hybrid methods. To the best of our knowledge, this is the first study that makes use of a voting mechanism to investigate the most relevant features. The remainder of the paper is organized as follows. In section 2, we briefly discuss related work. The evaluation dataset and related information is given in Section 3. In Section 4, we present ranker based filters and the machine learning algorithms used in the study. Section 5 gives details about proposed feature selection algorithm, detailed results of the conducted experiments, and ANOVA test employed for statistically validate the obtained results. In Section 6, validity threats of the study are presented. The article ends with conclusion and as well a list of references.

2. Related Work

There is an increasing effort on developing Search Based Software Engineering (SBSE) oriented algorithms for Software Product Lines (SPLs). SBSE addresses software engineering problems such as requirement analysis, predictive modeling, software project management, design, testing, refactoring and repair [9]. In the context of this

study, one of the most important fields of SBSE searches is related with the obtaining optimum feature model. In other words, a valuable search field in SBSE is to find alternative methods for selecting effective features. In this paper, as an answer to SBSE optimum feature model problem, we propose a hybrid feature selection strategy, i.e. MVFS, to investigate effective software metrics for fault-prediction [9, 10].

There are many feature selection methods used to obtain the most relevant subset of features particularly for improving defect discrimination performance of prediction algorithms [11]. Many studies from literature surveys feature selection strategies and in general feature selection algorithms are classified as filters, wrappers and hybrid methods [12]. Filter based feature methods makes ranking of features from the most relevant to the least relevant with the use of statistical and entropy-based correlation criteria [13]. Chi-square (CS), Gain Ratio (GR), Information Gain (IG), Symmetrical Uncertainty (SU) and ReliefF (RLF) are widely used feature ranker methods [14].

Though filter rankers are classifier independent feature selection methods, wrappers help to obtain relevant feature subset depending on the classification accuracy of a core classifier. The methods search whole feature space adding or removing features to calculate the estimated accuracy of the core classifier. Generally, an exhaustive search is impractical, and therefore non-exhaustive, search methods such as e genetic algorithms, greedy stepwise, best first or random search are often preferred. Since filtering based selection approaches are independent of a classifier they are more efficient from computational cost of view. However, this relative gain is obtained with the loss of awareness of possible dependency between features and the prediction algorithm [15].

The wrapper algorithms propose solution to take account this dependency while obtaining feature subset at the expense of a computational cost. Hybrid feature selection strategies are trade-off solutions for both feature selection domains. They have made combination of multi feature selection approaches to acquire the best feature subset. One particular benefit of hybrid solutions help the use of benefits of filter and wrapper approaches. For instance a combination of filter selection methods is used in [16] to obtain a promising feature subset. The authors have developed a hybrid similarity measure based on defect categories and compare the performance of their metric with IG, GR and RLF on Area Under the Curve (AUC) metric. They have made use of 11 NASA Promise datasets and they have obtained about 70 % better values in terms of number of projects with the use of their hybrid similarity measure in comparison to classical filtering approaches. In an another two-step hybrid feature selection strategy [17], the authors have used CS, SU and RLF to determine relevance of the software metrics in tandem with a clustering strategy to obtain the optimum subset of features from Eclipse and NASA KC1 projects. They have utilized AUC and F-measure metrics to evaluate their results and they have stated that their hybrid methodology has increased the fault prediction performance compared to relevancy measures alone. In the literature, there are many feature selection studies that makes a combination of filter and wrapper approaches in some way to obtain the most valuable feature subset. [18] is an empirical study that investigates value of hybrid feature selection strategies.

Having introduced the basis of general feature selection strategies, we now survey related work conducted in software defect prediction domain. One of the early studies in this context is the study conducted in [19]. The authors made use of filters ranker an empirical study to eliminate irrelevant features and they show that only a few software metrics are enough to build an effective defect predictor. In another feature selection study for software defect prediction, authors use an artificial immune system search for

building a wrapper model to evaluate a software fault predictor [20]. Jacob et al. propose a hybrid selection method combining information gain ratio and correlation based feature selection applied on NASA datasets [21]. In their detailed empirical work, Catal et al. investigate effects of various feature selection techniques on public datasets from PROMISE repository with the use of RF and NB algorithms [22]. In their recent work [23], the authors use a multivariate linear regression stepwise forward feature selection as a wrapper fashion to obtain optimal set of source code metrics. Another recent work makes use of a hybrid feature selection to improve fault prediction performance of machine learning algorithms [24]. As a last study from literature, Chen et al. improves performance of their machine learning algorithms with two-step hybrid feature selection methodology [25].

There are many studies in the software engineering domain making use of feature selection methods to improve defect prediction accuracies of the algorithms. One of the key points observed in the recent studies is that hybrid of feature selection strategies are preferred to take benefit of multiple extraction techniques at the same time. The rationale behind this approach is similar to ensemble learning methodologies that rely on the performance of ensemble learners rather than a single learner [26]. In this context, hybrid feature selection strategies are continuously explored particularly in software fault prediction domain. Our feature-selection combination strategy, i.e. MVFS, is explained in section 5.1 is an extension to the ongoing search.

3. Software Measurement Data

In this study, we have used datasets from PROMISE repository [27], Eclipse Equinox [28] and Eclipse JDT R3.1 [29] bug prediction datasets given in Table 1. First four datasets are from NASA software projects which were developed in C/C++ language for spacecraft instrument, storage management, flight and earth orbiting. Eclipse Equinox Bug Prediction Dataset which was developed in Java language for the infrastructure of the Java IDE. The brief descriptions of the datasets are presented in Table 1.

Table 1. The description of datasets

Dataset	Number of modules	Non-Defective	Defective	% Defect
CM1	498	449	49	9.83
JM1	10885	8779	2106	19.35
KC1	2109	1783	326	15.45
PC1	1109	1032	77	6.94
Eclipse Equinox	997	791	206	20.66
Eclipse JDT R3.1	3883	2611	1272	32.75

NASA datasets contain 22 attributes composed of 4 McCabe metrics [30], 9 base Halstead measures [31], 8 derived Halstead measures [32], and the last attribute is ‘defect’ with 2 classes (false or true, whether a software module is defective or not) [33]. The definition and description of these metrics are presented in Table 2.

Table 2. Description of NASA software metrics

Metric type	Software metrics	Description
McCabe	LOC	Line count of code
	v(g)	Cyclomatic complexity
	ev(g)	Essential complexity
	iv(g)	Design complexity
Derived Halstead	N	Total operators + operands
	V	Volume
	L	Program length
	D	Difficulty
	I	Intelligence
	E	Effort to write code
	B	Effort estimate
	T	Time estimator
Basic Halstead	IOCode	Line count
	IOComment	Comment count
	IOBlank	Blank line count
	IOCodeAndComment	Number of code and comment lines
	uniq_Op	Number of unique operators
	uniq_Opnd	Number of unique operands
	total_Op	Number of total operators
	total_Opnd	Number of total operands
	branchCount	Number of branch counts
Class	defects	Describing whether a software module is defective or not

On the other hand, Eclipse Equinox consists of 38 metrics: 6 Chidamber & Kemer (CK) metrics [34], 11 Object-Oriented (OO) metrics [35], 5 entropy metrics [35], 15 change metrics [36, 37], and the last metric is 'bug' that describing whether a file is bug or not. The brief description of these metrics is given in Table 3.

Table 3. Description of Eclipse Equinox software metrics

Metric type	Software metrics	Description
CK Metrics	WMC	Weighted method count
	DIT	Depth of inheritance tree
	RFC	Response for class
	NOC	Number of children
	CBO	Coupling between objects
	LCOM	Lack of cohesion in methods
OO Metrics	Fan-In	Number of other classes that reference the class
	Fan-Out	Number of other classes referenced by the class
	NOA	Number of attributes
	NOAI	Number of attributes inherited
	NOPA	Number of public attributes
	NOPRA	Number of private attributes
	LOC	Number of lines of code
	NOM	Number of methods

	NOMI	Number of methods inherited
	NOPM	Number of public methods
	NOPRM	Number of private methods
Entropy Metrics	HCM	Entropy of code changes
	WHCM	Weighted entropy
	LDHCM	Linearly decayed entropy
	LGDHCM	Logarithmically decayed entropy
	EDHCM	Exponentially decayed entropy
Change Metrics	NR	Number of revisions of a file
	NFIX	Number of times file was involved in bug-fixing
	NREF	Number of times file has been refactored
	NAUTH	Number of authors who committed the file
	LOC_ADDED	Sum over all revisions of the LOC added to a file
	maxLOC_ADDED	Maximum number of LOC added for all revisions
	avgLOC_ADDED	Average LOC added per revision
	LOC_REMOVED	Sum over all revisions of the LOC removed from a file
	max LOC_REMOVED	Maximum number of LOC removed for all revisions
	avg LOC_REMOVED	Average LOC removed per revision
	codeCHU	Sum of code churn over all revisions
	maxCodeCHU	Maximum code churn for all revisions
	avgCodeCHU	Average code churn per revision
	AGE	Age of a file in weeks
	WAGE	Weighted age
Class	Bugs	Describing whether a file is bug or not

The features of Eclipse JDT R3.1 dataset is taken from the study Mausa *et al.* [29].

4. Majority Vote Feature Selection Algorithm in Software Fault Prediction

In the literature, feature selection strategies are divided in three main groups, i.e. filters, wrappers, hybrid approaches, as aforementioned. The goal of the feature selection strategies is two-sided: (i) gain increase in the interpretability of the domain via decrease in feature space and (ii) obtain an improvement in the performances of the machine learning algorithms. With MVFS method, we explore a subset of software metrics that serve these two enhancements. In our method, we make use of combinations of filter approaches that are explained in the following sub-sections.

4.1. Feature Filtering Methods

4.1.1. Information Gain

Information Gain (IG) is a widely used feature selection method based on Shannon's entropy which describes the level of importance between random variable Y and a given information X [38]. In machine learning, IG is used to measure the attribute's information gain with respect to the class label. This method can be work with both nominal and numerical feature values with an appropriate normalization. IG score of an attribute A can be calculated as follows.

$$IG(A) = H(S) - \sum_i \frac{S_i}{S} H(S_i) \tag{1}$$

where H(S) is the total entropy of the dataset and $H(S_i)$ is the entropy of the ith subset generated by partitioning S based on feature A.

4.1.2. Symmetrical Uncertainty

Symmetrical Uncertainty (SU) is the normalized form of Information Gain [39] and is calculated with the following equation.

$$SU(S, A) = 2 * \frac{IG(S|A)}{H(S) + H(A)} \tag{2}$$

The SU method works similarly to IG. In addition to the score calculated for information gain, it defines the information content of a particular attribute, including definitions of the attribute and the entropy structure of the class.

4.1.3. ReliefF

ReliefF feature selection method measures the importance of an attribute by repeatedly sampling an instance and taking into account the value of the given attribute for its two nearest instances, one instance from the same class, and the other instance from the different class [40]. This method is very effective when working with large amounts of data. Since the number of performed sampling trials is constant, ReliefF feature selection method can run quicker than other methods. The algorithm of ReliefF method for a given m number of sampled instances and k number of features is shown in Figure 1.

```
Set all weights W[Ai] = 0.0;
for j = 1 to m do begin
  randomly select an instance X;
  find nearest hit H and nearest miss M;
  for I = 1 to k do
```

```
        W[Ai] = W[Ai] – diff(Ai, X, H) / m + diff(Ai, X, M) /
m
end;
```

Fig. 1. ReliefF Algorithm

4.1.4. Correlation based approach

Correlation based feature selection approach evaluates the importance of an attribute by measuring the Pearson's correlation between the attribute and the target class [41]. This method simply measures linear correlation between features. The following formula indicates the calculation of Correlation Coefficient (R) between the attribute A and class C.

$$R(f_i, y) = \frac{cov(f_i, y)}{\sqrt{var(f_i)\ var(y)}} \tag{3}$$

4.2. Machine Learning ClassifiersNaïve Bayes

Naïve Bayes (NB) is a well-known machine learning classifier based on statistical Bayes Theorem and conditional probability [42]. Bayes theorem provides to calculate the posterior probability, P(c | x), from P(c), P(x), and P(x | c). NB classifier presumes that the impact of the value of a feature (x) on a given class (c) is independent of the values of other attributes. This assumption is called class conditional independence and calculated with following equations.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{4}$$

$$P(c|x) = P(x_1|c) * P(x_2|c) * ... * P(x_n|c) * P(c) \tag{5}$$

where P(c | x) is the posterior probability of class given feature. P(c) is the prior probability of class. P(x | c) is the likelihood which is the probability of feature given class. P(x) is the prior probability of attribute.

4.2.2. Decision Tree

Decision tree is a supervised learning approach that classifies the test data by creating a flowchart-like decision tree based on a training set. In the constructed decision tree, internal nodes, branches, and leaflets indicate the features of dataset, values of features, and classification labels respectively. The main advantage to the use of decision trees is the class-oriented visualization of dataset. In this study, J48 decision tree learning algorithm which is a version of well-known Iterative Dichotomiser (ID) 3 is utilized [43].

4.2.3. K-Nearest Neighbor

K-Nearest Neighbor (K-NN) is a simple instance-based and lazy learning classification algorithm having no training phase [44]. The distance between the test data and remaining instances is calculated, finally the class having maximum count is selected from the nearest k samples. In K-NN, Euclidean and Cosine similarity measures are the most common algorithms to calculate the distance [45]. In the proposed study, a Weka implementation of K-NN algorithm called IBk is employed.

4.2. Evaluation Criteria

Different criteria are employed to evaluate the performance of classifiers in Machine Learning. All criteria are formulized using a confusion matrix that contains actual and predicted class labels. True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) indicates the four different prediction outcomes [46]. In software fault prediction literature, Geometric Mean - 1 (GM) is used by researchers such as Ma et al[46] and Cagatay et al[47] for the valuation of prediction systems to benchmark ML algorithms [48]. In this study, we therefore have used GM to evaluate performance of our algorithms. GM is also a good performance indicator when the datasets are imbalanced and it is used for the evaluation of fault prediction systems [47]. GM metric is calculated using Eq. 6.

$$\text{Geometric Mean1} = \sqrt{(precision * recall)} \tag{6}$$

In 6, Precision is the ratio of correctly predicted positive instances and total predicted positive instances. Furthermore Recall is the defined as the ratio of correctly predicted positive instances and total number of correctly observed positive instances. Precision and Recall are calculated with Eq. 7 and 8 respectively.

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

5. Experimental Study and Analysis

5.1. Design

In this section, we supply a detailed pseudocode that explains the details of MVFS algorithm.

Input:

D: array [KC1, PCI, JM1, CM1, EclipseCore, EclipseJDT]: Datasets
S: Full feature set of selected dataset
MLA: array [IBK, NB, J48]: Machine learning algorithms
SMLA: Selected machine learning algorithm
Rt array [IG, CO, RF, SU]: Feature selection techniques
r: Number of ranker
voting: Number of voting

Output:

o_1: Specifies Gm (G-means)

Method:

(1) S = D[Selected data set] // A dataset is selected.
(2) o_1 = MLA[smla(S)], o_2= MLA[smla(S)] // Selected machine learning algorithms are trained and tested.
Gm = o_1 // The value of o_1 measure is obtained.
(3) S = D[Selected data set] // Dataset is selected once again for feature selection.
(4) o_1 = Rt(MLA[smla(S)], r), o_2 = Rt(MLA[smla(S)], r)
Gm = o_1 // Gm performance metric is obtained.
(5) while (Rt((S), r) for each dataset)
{
if (voting = "2/4") // Voting count is identified.
voting_count=2
else if (voting = "3/4")
voting_count=3
else if (voting = "4/4")
voting_count=4
}
(6) o_1 =MLA[smla(voting(S), r)] // 3 new datasets for each voting rule are obtained and machine learning algorithms trained and tested using cross validation.
Gm= o_1 // Gm performance metric for all generated datasets is obtained.

Fig. 2. The pseudocode of MVFS Algorithm.

In brief terms, the proposed algorithm runs as follows:

NASA and Eclipse projects are used with all features and tested with NB, J48 and IBK algorithms on top of 10-fold cross validation scheme to obtain GM metric. In the second phase IG, CO, RF and SY rankers are used to obtain top 20 software metrics and the experiments are revaluated. In this phase, MVFS algorithm is run using 2/4, 3/4 and 4/4 voting rules and 3 new datasets with reduced features are obtained. The dimensionally reduced datasets are used and GM metric is obtained for NB, J48 and IBK classifiers. This cycle is repeated as follows: (i) obtain subset of features gradually 20, 15, 10, 5 and run MVFS to obtain 3 new data sets for each voting rule, (ii) use 10-CV train-test model for NB, J48 and IBK and obtain GM performance metric for all generated datasets.

All the runs are performed using the implementations of NB, J48, and IBK algorithms in the WEKA (Waikato Environment for Knowledge Analysis) version 3.8.1 [49]. The default parameters are used for each algorithm and the mentioned ranker methods since they produce promising results as stated in [50]. For NB having

continuous variables, any kernel method for prediction of the distribution is not used. The default parameters for IBK and J48 algorithm are also employed in the study. For IBK, the value of parameter k is selected as 1, distance weighting is not applied and Euclidean distance is chosen as distance function.

5.2. Results

In this section, we give details corresponding to the designed algorithm. The overall results corresponding to each project are given in related tables. However, we did not provide the results for all voting rules. We instead selected the best performance metrics for the sake of convenience. Additionally, in order to make interpretation of tables easier and illustrate the performance of the proposed algorithm more obvious, we produced recapping figures for each table.

Table 4. Experimental Results for Project KC1

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.838	0.839	0.829	0.832	0.810
	J48	0.835	0.839	0.830	0.833	0.837
	NB	0.819	0.819	0.823	0.824	0.827
CO	IBk	0.838	0.839	0.831	0.828	0.828
	J48	0.835	0.840	0.835	0.834	0.833
	NB	0.819	0.819	0.821	0.823	0.825
RF	IBk	0.838	0.838	0.832	0.839	0.831
	J48	0.835	0.833	0.835	0.833	0.836
	NB	0.819	0.819	0.819	0.818	0.816
SY	IBk	0.838	0.839	0.835	0.825	0.822
	J48	0.835	0.840	0.836	0.825	0.828
	NB	0.819	0.819	0.825	0.823	0.824
FS Method	Classifier Name		19 out of 20	8 out of 15	2 out of 10	1 out of 5
		GM	GM	GM	GM	GM
Majority Vote	IBk	0.838	0.839	0.833	0.826	0.829
	J48	0.835	0.840	0.839	0.835	0.834
	NB	0.819	0.819	0.822	0.823	0.837

In Table 4, it is seen that MVFS yields acceptable values in terms of GM compared to the standard feature ranker algorithms. The results may be examined in Figure 3 more precisely.

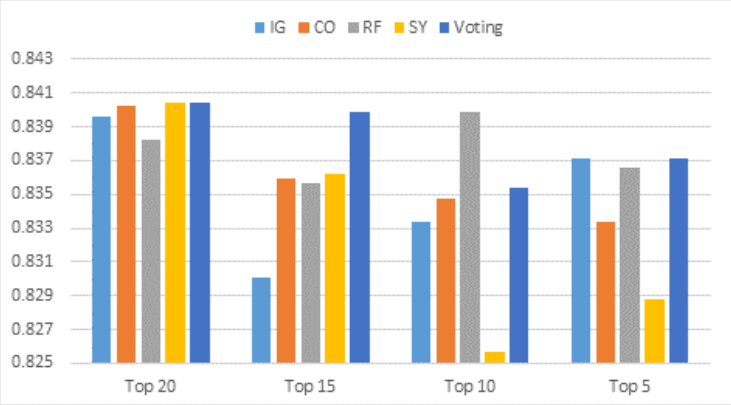


Fig. 3. Illustration of Experimental Results of Table 4

Table 5. Experimental Results for Project PC1

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.921	0.919	0.912	0.916	0.913
	J48	0.925	0.921	0.917	0.919	0.934
	NB	0.895	0.894	0.897	0.894	0.895
CO	IBk	0.921	0.919	0.916	0.910	0.915
	J48	0.925	0.921	0.920	0.923	0.922
	NB	0.895	0.894	0.896	0.897	0.895
RF	IBk	0.921	0.921	0.915	0.917	0.920
	J48	0.925	0.923	0.918	0.921	0.932
	NB	0.895	0.894	0.893	0.892	0.896
SY	IBk	0.921	0.919	0.915	0.916	0.912
	J48	0.925	0.921	0.920	0.919	0.925
	NB	0.895	0.894	0.895	0.894	0.900
FS Method	Classifier Name		19 out of 20	11 out of 15	10 out of 10	2 out of 5
		GM	GM	GM	GM	GM
Majority Vote	IBk	0.921	0.919	0.915	0.916	0.916
	J48	0.925	0.921	0.920	0.923	0.935
	NB	0.895	0.894	0.895	0.902	0.907

While examining Table 5 for the experiments of project PCI, we may observe that MVFS technique improves the performance of the algorithms particularly for top 5 software metrics. This improvement is observed in Figure 4 clearly.

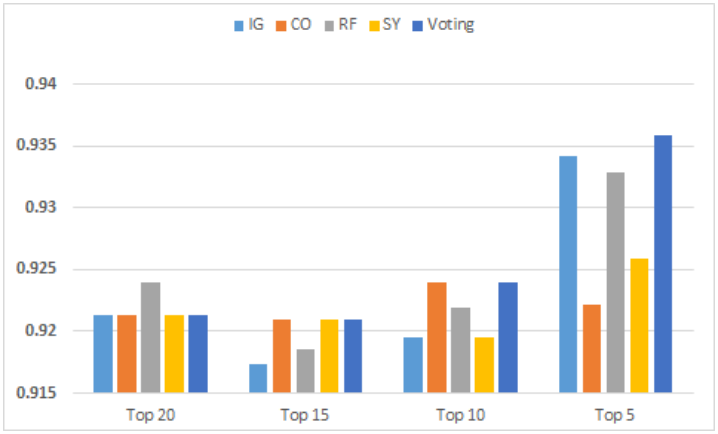


Fig. 4. Illustration of Experimental Results of Table 5

The experiment conducted on Project JM1 is given in Table 6 and the visualization of the results is provided in Figure5.

Table 6. Experimental Results for Project JM1

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.766	0.765	0.762	0.761	0.758
	J48	0.776	0.782	0.787	0.792	0.792
	NB	0.784	0.784	0.783	0.782	0.782
CO	IBk	0.766	0.766	0.762	0.759	0.764
	J48	0.776	0.776	0.785	0.787	0.790
	NB	0.784	0.783	0.783	0.784	0.787
RF	IBk	0.766	0.766	0.766	0.764	0.755
	J48	0.776	0.777	0.781	0.789	0.784
	NB	0.784	0.783	0.783	0.784	0.774
SY	IBk	0.766	0.765	0.761	0.760	0.752
	J48	0.776	0.783	0.791	0.786	0.798
	NB	0.784	0.784	0.783	0.781	0.785
FS Method	Classifier Name		18 out of 20	9 out of 15	7 out of 10	5 out of 5
		GM	GM	GM	GM	GM
Majority Vote	IBk	0.766	0.766	0.763	0.786	0.775
	J48	0.776	0.782	0.791	0.809	0.788
	NB	0.784	0.784	0.784	0.785	0.787

As the Table 6 and Figure 5 is evaluated together, top 10 software metrics obtained with MVFS method result in higher GM values.

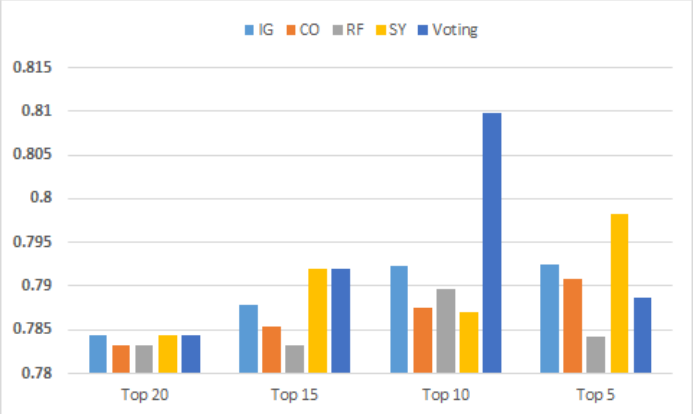


Fig. 5. Illustration of Experimental Results of Table 6

CM1, another NASA projects, is evaluated with the same scheme explained before. The results of the experiments and the corresponding sum up figure is given as Table 7, Figure 6 respectively.

Table 7. Experimental Results for Project CM1

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.843	0.843	0.851	0.835	0.848
	J48	0.855	0.850	0.851	0.850	0.850
	NB	0.857	0.856	0.856	0.862	0.866
CO	IBk	0.843	0.843	0.836	0.837	0.832
	J48	0.855	0.855	0.859	0.850	0.851
	NB	0.857	0.857	0.855	0.857	0.866
RF	IBk	0.843	0.843	0.838	0.847	0.830
	J48	0.855	0.854	0.853	0.845	0.856
	NB	0.857	0.857	0.855	0.855	0.855
SY	IBk	0.843	0.843	0.851	0.835	0.840
	J48	0.855	0.855	0.851	0.850	0.851
	NB	0.857	0.857	0.856	0.862	0.870
FS Method	Classifier Name		19 out of 20	13 out of 15	9 out of 10	4 out of 5
		GM	GM	GM	GM	GM
Majority Vote	IBk	0.843	0.843	0.848	0.851	0.861
	J48	0.855	0.855	0.856	0.850	0.856
	NB	0.857	0.857	0.855	0.869	0.872

From Table 7 and Figure 6, we may observe that, the proposed method does not improve the results at first glance. However, it retains the classification performance metrics at top 10 and top 5 features.

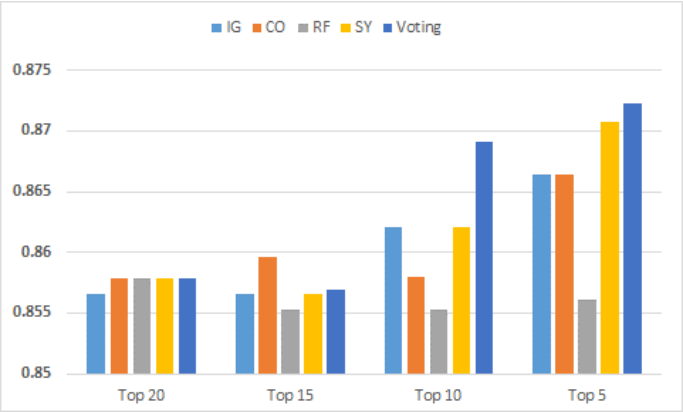


Fig. 6. Illustration of Experimental Results of Table 7

The evaluation results of Eclipse Equinox dataset, is provided in Table 8 and corresponding Figure 7.

Table 8. Experimental Results for Project Eclipse Equinox Core Dataset

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.799	0.819	0.804	0.804	0.801
	J48	0.805	0.824	0.823	0.818	0.845
	NB	0.827	0.833	0.841	0.840	0.840
CO	IBk	0.799	0.808	0.805	0.797	0.775
	J48	0.805	0.812	0.802	0.820	0.832
	NB	0.827	0.827	0.839	0.840	0.840
RF	IBk	0.799	0.787	0.788	0.802	0.743
	J48	0.805	0.806	0.811	0.805	0.758
	NB	0.827	0.843	0.839	0.829	0.754
SY	IBk	0.799	0.816	0.809	0.811	0.794
	J48	0.805	0.814	0.819	0.830	0.834
	NB	0.827	0.835	0.841	0.843	0.838
FS Method	Classifier Name		18 out of 20	13 out of 15	10 out of 10	3 out of 5
		GM	GM	GM	GM	GM
Majority Vote	IBk	0.799	0.810	0.812	0.804	0.798
	J48	0.805	0.825	0.828	0.848	0.842
	NB	0.827	0.838	0.842	0.840	0.848

The results of the experiments show a fairly increase in GM metrics, in particular at top 10 and top 5 software metrics selected with MVFS.

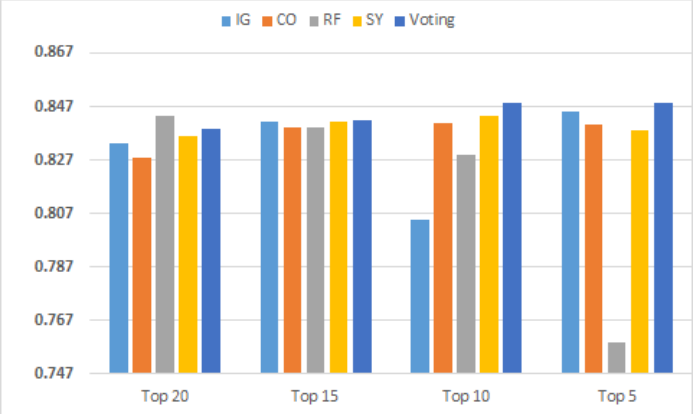


Fig. 7. Illustration of Experimental Results of Table 8

The evaluation results of last dataset, Eclipse JDT dataset, is provided in Table 9 and in the corresponding Figure 8.

Table 9. Experimental Results for Project Eclipse JDT Dataset

FS Method	Classifier Name	All Features GM	Top 20 Features GM	Top 15 Features GM	Top 10 Features GM	Top 5 Features GM
IG	IBk	0.709	0.704	0.708	0.692	0.698
	J48	0.721	0.722	0.719	0.744	0.734
	NB	0.720	0.719	0.716	0.720	0.720
CO	IBk	0.709	0.721	0.707	0.691	0.686
	J48	0.721	0.729	0.726	0.732	0.732
	NB	0.720	0.719	0.722	0.723	0.722
RF	IBk	0.709	0.703	0.704	0.687	0.674
	J48	0.721	0.729	0.727	0.716	0.691
	NB	0.720	0.723	0.720	0.725	0.683
SY	IBk	0.709	0.711	0.708	0.689	0.695
	J48	0.721	0.716	0.718	0.739	0.737
	NB	0.720	0.718	0.716	0.719	0.716
FS Method	Classifier Name		17 out of 20 GM	12 out of 15 GM	8 out of 10 GM	1 out of 5 GM
Majority Vote	IBk	0.709	0.718	0.709	0.684	0.693
	J48	0.721	0.732	0.737	0.739	0.738
	NB	0.720	0.720	0.720	0.722	0.723

The results of the experiments show a considerable increase in GM metrics, in particular at top 20, top 15 and top 5 software metrics selected with MVFS.

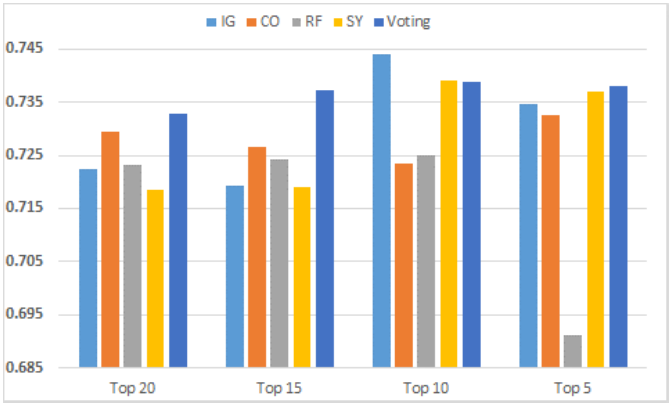


Fig. 8. Illustration of Experimental Results of Table 9

As a sum up of this section, we summarize the results of the experiments and compare the performance of our method with conventional feature rankers based on GM metric in Table 10.

Table 10. Overall Evaluation of Experimental Results based on GM

	Top20	**Top 15**	**Top 10**	**Top 5**
KC1	same	**better**	worse	same
PC1	worse	same	same	**better**
JM1	same	same	**better**	worse
CM1	same	worse	**better**	**better**
Eclipse Equinox Core	worse	same	**better**	**better**
Eclipse JDT	**better**	**better**	worse	**better**

As the Table 10 is examined, it is seen that the proposed method is eligible to discriminate most valuable software metrics that are functional in software fault prediction detection. Table 10 provides the comparative results of proposed MVFS algorithm and standard rankers, i.e., IG, SU, RF and CO.

For all datasets, MVFS algorithm yields similar results compared to conventional rankers for top 20 and top 15 features. Furthermore, as it can be observed from Table 10, our approach is able to find the most informative software metrics at top 10 or top 5 features. As an overall summary, we may draw a conclusion from Table 10 that the proposed method either increases the prediction of the algorithms or keep their performance as the same.

We have moreover calculated mean and medians of the predictions of the classifiers from the related tables to compare overall results. The results of these statistical calculations are given in Table 11.

As the Table 11 is inspected with median and mean perspectives, it can easily be observed that the proposed method is almost better than the remaining algorithms in terms of fault prediction.

Table 11. Median and Mean Calculations of Classier Results based on GM

		Top20	Top 15	Top 10	Top 5
KC1	Mean Value	0.839	0.834	0.831	0.834
	Median Value	0.839	0.835	0.833	0.834
	Majority Value	0.840	0.839	0.835	0.837
	Median Base Result	**better**	**better**	**better**	**better**
	Mean Base Result	**better**	**better**	**better**	**better**
PC1	Mean Value	0.921	0.918	0.920	0.928
	Median Value	0.921	0.919	0.920	0.928
	Majority Value	0.921	0.920	0.923	0.935
	Median Base Result	same	**better**	**better**	**better**
	Mean Base Result	same	**better**	**better**	**better**
JM1	Mean Value	0.783	0.786	0.788	0.791
	Median Value	0.784	0.786	0.788	0.791
	Majority Value	0.784	0.791	0.809	0.788
	Median Base Result	**better**	**better**	**better**	worse
	Mean Base Result	same	**Better**	**better**	worse
CM1	Mean Value	0.856	0.856	0.859	0.864
	Median Value	0.857	0.856	0.859	0.866
	Majority Value	0.857	0.856	0.869	0.872
	Median Base Result	**better**	same	**better**	**better**
	Mean Base Result	same	same	**better**	**better**
Eclipse Equinox Core	Mean Value	0.834	0.840	0.838	0.820
	Median Value	0.834	0.840	0.840	0.839
	Majority Value	0.838	0.842	0.848	0.848
	Median Base Result	**better**	**better**	**better**	**better**
	Mean Base Result	**better**	**better**	**better**	**better**
Eclipse JDT	Mean Value	0.724	0.722	0.735	0.723
	Median Value	0.725	0.722	0.735	0.733
	Majority Value	0.732	0.737	0.739	0.738
	Median Base Result	**better**	**better**	**better**	**better**
	Mean Base Result	**better**	**better**	**better**	**better**

5.3. ANOVA Test and Validation

Analysis of variance (ANOVA) test was used to statistically validate the results of empirical analysis. In this study, two-way ANOVA test is employed to determine whether the differences between multiple groups of results are statistically significant based on independent factors. In the ANOVA test, feature selection methods, classifiers, feature sets and datasets are taken as the factors of the analysis. In addition, the interactions (interactions through order 2) between different factors are also taken into consideration. Namely, feature selection method and classifier interaction, feature selection method and feature set interaction, feature selection method and dataset interaction, classifier and feature set interaction and classifier and dataset interaction are also considered. In the analysis, a GM values are taken as the response values. The

hypothesis regarding the method performed on Minitab statistical software. The statistical values of ANOVA test are given in Figure 9. Parameters of the test, namely, DF, SS, MS, F and p-value correspond to degrees of freedom, adjusted sum of squares, adjusted mean square, F-statistics and probability value, respectively [51].

Analysis of Variance (GM)

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Classifier	2	0.01427	0.007136	122.29	0.000
Feature Selection Method	4	0.00373	0.000933	15.99	0.000
Feature Set	4	0.00048	0.000120	2.06	0.086
Dataset	5	1.57198	0.314397	5388.24	0.000
Classifier*Feature Selection Method	8	0.00024	0.000030	0.52	0.842
Classifier*Feature Set	8	0.00309	0.000387	6.63	0.000
Classifier*Dataset	10	0.02969	0.002969	50.88	0.000
Feature Selection Method*Dataset	20	0.00444	0.000222	3.81	0.000
Feature Set*Dataset	20	0.00276	0.000138	2.36	0.001
Error	350	0.02042	0.000058		
Total	431	1.68526			

Fig. 9. Two-way ANOVA test results

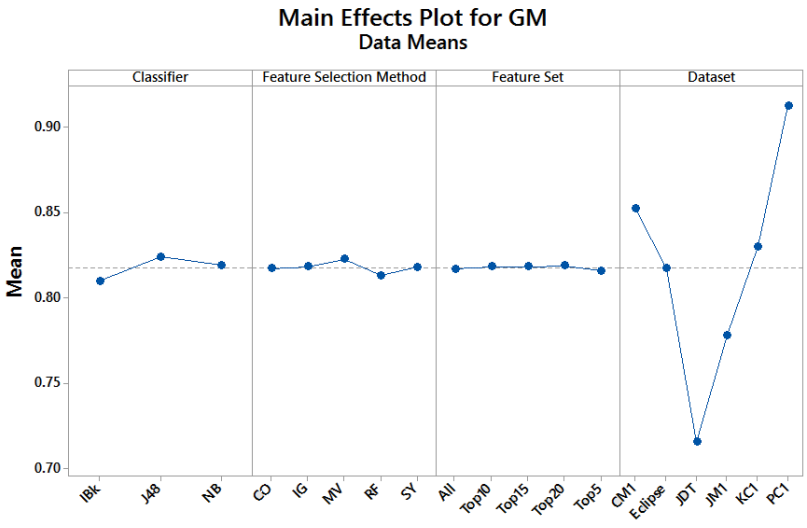


Fig. 10. Main Effects Plot for G-means

As indicated in Figure 9, the better predictive performance obtained by the proposed new feature selection method is statistically significant at 99% confidence level. Regarding the test results presented in Figure 9, there are statistically significant differences among the factors of the analysis (such as Classifier, Feature selection method, Dataset, etc.). Hence, it can be seen clearly in Figure 9 that most of the values obtained are statistically significant at 99% confidence level. However, there is no

statistically significant difference between the GM values for feature selection method and classifiers. That is, differences for predictive performance of different feature sets do not exhibit a varying pattern based on the classifiers.

In Figure 10, the main effects plots for classification GM values of the empirical analysis is given. It summarizes comparatively the main findings of the study based on the average results of experiments. As it can be seen from Figure 10, the highest performance in terms of accuracy values were obtained by the proposed majority voting based feature section method. Regarding the performance of classification algorithms, the highest predictive performance (in terms of GM) was achieved by J48 algorithm. Regarding the datasets utilized in the empirical analysis, PC1 dataset yields the highest performance. In Figure 11, the histograms of residuals for all empirical results (in terms of GM) are presented to examine the distribution of empirical results. As it can be observed, the patterns for residuals of all observations exhibit a skewed distribution, which validate the statistical differences obtained by two-way ANOVA test results presented in Figure 11.

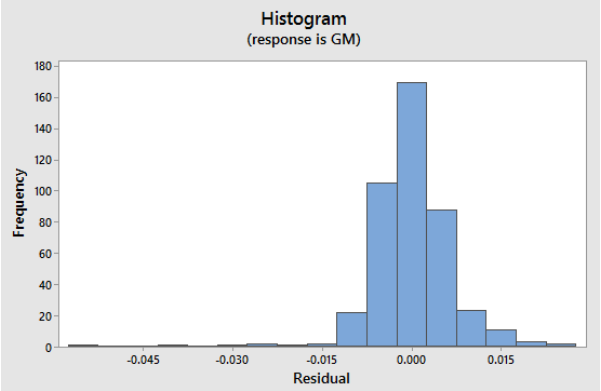


Fig. 31. Histogram of Residuals

6. Threads to Validity

This empirical study uses six bug prediction datasets. Four from PROMISE repository, and two from Eclipse domain. One of the main threads of such empirical studies is that the generalization capability of the methods may be insufficient and domain dependent. Naturally, the proposed feature selection method may provide varying results in different software domains. Furthermore, the success of empirical software analysis strategies is highly dependent on the quality of selected metrics. The main emphasis of this study is to develop an ensemble feature selection strategy that combines various selection algorithms to obtain the best feature subset. Being a pre-processing step, the selection of classifiers probably has minor influence on the final decisions. Though not guaranteed, this feature selection strategy is expected to provide better results compared with the results of a single feature selection method. Finally, the significance of the proposed method is supported with statistical tests.

Selection of classifier parameters, being another thread, have also influence on the corresponding results. Even the same classifier with different parameters may affect the related fault-detection performance and therefore selection of the set of classifier parameters are also evaluated in the study.

One more important thread is that the classifier fault-detection performances may vary with the selected subset of metrics. In this context, our proposed method uses a voting ensemble strategy to obtain an optimal set of metrics. The method makes use of a majority combination rule to select the most significant metrics from the results of four different widely used ranker algorithms. Voting based ensembles may also be obtained with averaging or weighting combination mechanisms that may influence the results. Other empirical studies may take advantage of various combinations to obtain the best subset of software metrics [52].

This study makes use of various feature selection algorithms, their ensemble combinations. The quality of the obtained subset of features is evaluated with various classifiers. To reduce modeling errors to minimum, the experiments and statistical investigation were conducted by only one skilled researcher.

7. Conclusion

Quality in selection of software metrics is critical in the fault detection performance of prediction models. Therefore intelligent selection of software metrics is the first step to obtain an accurate model. We present a collaborative feature selection model to discriminate the most informative software metrics and eliminate other irrelevant metrics. We made use of six software projects and three machine learning algorithms in order to compare performance of or MVFS algorithm with four conventional rankers. As a result, it is empirically observed in the sum up Table 10 that the proposed method is either increases the fault detection performance or retain it as the same compared to the performance of the standard feature selection methods. The obtained experimental results are statistically supported with two-way ANOVA test conducted for GM values. As a future work, we want to test the performance of the proposed MVFS algorithm with the use of another software projects to demonstrate its efficiency.

References

1. Koru, A. G., Liu, H.: Building Effective Defect-Prediction Models in Practice. IEEE Software. 22(6): 23-29. (2005)
2. Azeem, N., Usmani, S.: Defect Prediction Leads to High Quality Product. Journal of Software Engineering and Applications. 4(11): 639-645. (2011)
3. Gao, K., Khoshgoftaar, T. M., Wang, H., Seliya, N.: Choosing software metrics for defect prediction: an investigation on feature selection techniques. Software Practice and Experience. 41(5): 579-606. (2011)
4. Wang, H., Khoshgoftaar, T. M., Gao, K., Seliya, N.: Mining Data from Multiple Software Development Projects. IEEE International Conference on Data Mining Workshops (ICDMW '09). (2009)

5. Mandal, P., Ami, A. S.: Selecting Best Attributes for Software Defect Prediction. IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE). (2015)
6. Khoshgoftaar, T. M., Gao, K., Napolitano, A.: An Empirical Study of Predictive Modeling Techniques of Software Quality. In: Suzuki J., Nakano T. (eds.) Bio-Inspired Models of Network, Information, and Computing Systems. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. vol. 87. Springer. Berlin, Heidelberg, pp. 288-302. (2012)
7. Onan, A., Korukoğlu, S., Bulut, H.: A multiobjective weighted voting ensemble classifier based on differential evolution algorithm for text sentiment classification. Expert Systems with Applications. 62(C): 1-16. (2016)
8. Witten, I. H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. 2nd edition. Morgan Kaufmann. (2005)
9. Harman, M., Jia, Y., Krinke, J., Langdon, W. B., Petke, J., Zhang, Y.: Search based software engineering for software product line engineering: a survey and directions for future work. Proceedings of the 18th International Software Product Line Conference. vol. 1. pp. 5-18. (2014)
10. Shahpar, Z., Khatibi, V., Tanavar, A., Sarikhani, R.: Improvement of effort estimation accuracy in software projects using a feature selection approach. Journal of Advances in Computer Engineering and Technology. 2(4): 31-38. (2016)
11. Chu, C., Hsu, A-L., Chou, K-H., Bandettini, P., Lin, C-P.: Does feature selection improve classification accuracy? Impact of sample size and feature selection on classification using anatomical magnetic resonance images. NeuroImage. 60(1): 59-70. (2012)
12. Jović, A., Brkić, K., Bogunović, N.: Review of feature selection methods with applications. 38th International Convention on Information and Communication Technology. Electronics and Microelectronics (MIPRO). (2015)
13. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research. vol. 3. pp. 1157-1182. (2003).
14. Novaković, J., Strbac, P., Bulatović, D.: Toward Optimal Feature Selection Using Ranking Methods and Classification Algorithms. Yugoslav Journal of Operations Research. 21(1): 119-135. (2011).
15. Janecek, A. G. K., Gansterer, W. N., Demel, M. A., Ecker, G. F.: On the Relationship between Feature Selection and Classification Accuracy. Proceedings of the 2008 International Conference on New Challenges for Feature Selection in Data Mining and Knowledge Discovery. vol. 4. pp. 90-105. (2008)
16. YU .Q., JIANG S., WANG R., WANG H. : A feature selection approach based on a similarity measure for software defect prediction. Frontiers of Information Technology & Electronic Engineering. pp. 1744-1753 (2017)
17. Chen J., Liu S., Chen X., Gu Q., Chen D. Empirical Studies on Feature Selection for Software FaultPrediction, 5th Asia-Pacific Symposium on Internetware, (2013)
18. Afzal, W., Torkar, R.: Towards Benchmarking Feature Subset Selection Methods for Software Fault Prediction. Computational Intelligence and Quantitative Software Engineering, pp. 33-58. (2016)
19. Wang, H., Khoshgoftaar, T. M., Seliya, N.: How Many Software Metrics Should be Selected for Defect Prediction?. Proceedings of the Twenty-Fourth International Florida Artificial Intelligence Research Society Conference. Palm Beach, Florida, USA. (2011)
20. Soleimani, A., Asdaghi, F.: An AIS based feature selection method for software fault prediction. Iranian Conference on Intelligent Systems. IEEE. (2014)

21. Jacob, S., Raju, G.: Software Defect Prediction in Large Space Systems through Hybrid Feature Selection and Classification. The International Arab Journal of Information Technology, 14(2): 208-214. (2017)
22. Catal, C., Diri, B.: Investigating the effect of dataset size, metrics sets, and feature selection techniques on software fault prediction problem, Information Sciences, 179(8): 1040-1058. (2009)
23. Kumar, L., Rath, S., Sureka, A.: Using Source Code Metrics and Ensemble Methods for Fault Proneness Prediction. https://arxiv.org/abs/1704.04383, (2017)
24. Alighardashi, F., Chahooki, M. A. Z.: The Effectiveness of the Fused Weighted Filter Feature Selection Method to Improve Software Fault Prediction. Journal of Communications Technology. Electronics and Computer Science. vol. 8. pp. 5-11. (2016)
25. Chen, J., Liu, S., Chen, X., Gu, Q., Chen, D.: Empirical Studies on Feature Selection for Software Fault Prediction. Proceedings of the 5th Asia-Pacific Symposium on Internetware. (2013)
26. Dietterich, T. G.: Ensemble Methods in Machine Learning. In: Multiple Classifier Systems. Lecture Notes in Computer Science. vol. 1857. Springer, Berlin, Heidelberg, pp. 1-15. (2000)
27. Software Defect Dataset, Promise Repository. January (2018). [Online]. Available: http://promise.site.uottawa.ca/SERepository/datasets-page.html
28. Eclipse Bug Prediction Dataset**.** Eclipse Equinox. January (2018). [Online]. Available: http://bug.inf.usi.ch/data/eclipse.zip
29. Mauša G., Grbac T. G., Bašić B. D., A Systematic Data Collection Procedure for Software Defect Prediction, ComSIS Consortium vol 13, Issue 1 , (2016)
30. McCabe, T. J.: A complexity measure. IEEE Transactions on Software Engineering. 2(4): 308–320. (1976)
31. Halstead, M. H.: Elements of software science. Elsevier Computer Science Library. Operating and Programming Systems Series. New York, Oxford, (1977)
32. Tutorial on McCabe and Halsted. January (2018). [Online]. Available: http://openscience.us/repo/defect/mccabehalsted/tut.html
33. Rodriguez, D., Ruiz, R., Gallego, J. C., Aguilar-Ruiz, J.: Detecting Fault Modules Applying Feature Selection to Classifiers. IEEE International Conference on Information Reuse and Integration. (2007)
34. Basili, V. R., Briand, L. C., Melo, W. L.: A Validation of Object-Oriented Design Metrics as Quality Indicators. IEEE Transactions on Software Engineering. 22(10): 751-761. (1996)
35. D'Ambros, M., Lanza, M., Robbes, R.: An Extensive Comparison of Bug Prediction Approaches. 7th IEEE Working Conference on Mining Software Repositories. IEEE CS Press. pp. 31-41. (2010)
36. Moser, R., Pedrycz, W., Succi, G.: A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction. Proceedings of the 30th International Conference on Software Engineering. pp. 181–190. (2008)
37. Muthukumaran, K., Bhanu Murthy, N. L.: Impact of Restricted Forward Greedy Feature Selection Technique on Bug Prediction. PeerJ Computer Science. (2015)
38. Yang, Y., Pedersen, J. O.: A comparative study on feature selection in text categorization. Proceedings of the 14th International Conference on Machine Learning (ICML '97). Nashville, Tenn, USA. Morgan Kaufmann. pp. 412–420. (1997)
39. Press, W. H., Teukolsky, S. A., Vetterling, W. T., Flannery, B. P.: Numerical Recipes in C. Cambridge University Press. (1988)

40. Kononenko, I.: Estimating Attributes: Analysis and Extensions of RELIEF. In: Proceedings of the 7th European Conference on Machine Learning, pp. 171–182. (1994)
41. Hall, M. A., Smith, L. A.: Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper. Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference. pp. 235-239. (1999)
42. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research. vol. 12. pp. 2825-2830. (2011)
43. Khoshgoftaar, T. M., Seliya, N.: Tree-based software quality estimation models for fault prediction. Eighth IEEE Symposium on Software Metrics. pp. 203-214. (2002)
44. Aha, D. W., Kibler, D., Albert, M. K.: Instance-based Learning Algorithms. Machine Learning. 6(1): 37-66. (1991)
45. Qian, G., Sural, S., Gu, Y., Pramanik, S.: Similarity between Euclidean and cosine angle distance for nearest neighbor queries. Proceedings of the 2004 ACM symposium on applied computing. pp. 1232-1237. (2004)
46. Ma Y., Guo L., Cukic B., A Statistical Framework for the Prediction of Fault-Proneness, Advances in Machine Learning Application in Software Engineering, Idea Group Inc., pp. 237-265, (2006)
47. Catal C., Performance Evaluation Metrics for Software Fault Prediction Studies, Acta Polytechnica Hungarica vol. 9, no. 4, (2012)
48. Y. Ma, L. Guo, B. Cukic, A Statistical Framework for the Prediction of Fault-Proneness, Advances in Machine Learning Application in Software Engineering, Idea Group Inc., 2006, pp. 237-265
49. Witten IH., Frank E., Data Mining: Practical Machine Learning Tools and Techniques. San Francisco, CA: Morgan Kaufman, (2005)
50. Amancio DR., Comin CH., Casanova D.,Travieso G.,Bruno OM.,Rodrigues FA.,Costa L., A systematic comparison of supervised classifiers. PloS one; 9(4): pp. 94-137 (2014).
51. Onan, A., Korukoglu, S., Bulut, H.: Ensemble of keyword extraction methods and classifiers in text classification. Expert Systems with Applications. 57(C): 232–247. (2016)
52. Polikar R., Ensemble Based Systems in Decision Making IEEE Circuits And Systems Magazinepp 21-45 (2006).

Emin Borandag is currently working as an Assistant Professor in the Department of Software Engineering, Manisa Celal Bayar University, Manisa, Turkey. He received BS and MS degrees in computer engineering from Maltepe University in 2003 and 2006, respectively. He was awarded PhD in computer engineering from Trakya University in 2011. His research interests are the area of software quality assurance, software testing and data mining.

Akin Ozcift is currently working as an Associate Professor in the Department of Software Engineering, Celal Bayar University, Manisa, Turkey. He received BS and MS degrees in Electronics Engineering from METU and Hacettepe Engineering in 1995 and 2000, respectively. He received PhD in Electronics Engineering from Firat University in 2011. His research interests focus on Machine Learning, Cyber Security and Big Data analysis.

Deniz Kilinc is currently working as an Assoc Professor in the Department of Software Engineering, Celal Bayar University, Manisa, Turkey. He received BS and MS degrees in computer engineering from Dokuz Eylül University in 2002 and 2004, respectively. He was awarded PhD in computer engineering from Dokuz Eylül University in 2010. His research interests focus on information retrieval, software engineering, text mining and social media
analysis.

Fatih Yücalar is currently working as an Assistant Professor in the Department of Software Engineering, Manisa Celal Bayar University, Manisa, Turkey. He received BS and MS degrees in computer engineering from Maltepe University in 2002 and 2006, respectively. He was awarded PhD in computer engineering from Trakya University in 2011. His research interests focus on software engineering, software project management, software quality assurance and testing.

Received: March 12, 2018; Accepted: December 31, 2018

Computer Science and Information Systems 16(2):541–564 https://doi.org/10.2298/CSIS180608009L

Reducing energy usage in resource-intensive Java-based scientific applications via micro-benchmark based code refactorings*

Mathias Longo[1], Ana Rodriguez[2], Cristian Mateos[2], and Alejandro Zunino[2]

1 University of Southern California,
1337 1/2 W Adams Blvd, Los Angeles (90007), United States
mathiasl@usc.edu
2 ISISTAN-CONICET-UNICEN,
Campus Universitario, Tandil (B7001BBO), Argentina
{ana.rodriguez,cristian.mateos,alejandro.zunino}@isistan.unicen.edu.ar

Abstract. In-silico research has grown considerably. Today's scientific code involves long-running computer simulations and hence powerful computing infrastructures are needed. Traditionally, research in high-performance computing has focused on executing code as fast as possible, while energy has been recently recognized as another goal to consider. Yet, energy-driven research has mostly focused on the hardware and middleware layers, but few efforts target the application level, where many energy-aware optimizations are possible. We revisit a catalog of Java primitives commonly used in OO scientific programming, or micro-benchmarks, to identify energy-friendly versions of the same primitive. We then apply the micro-benchmarks to classical scientific application kernels and machine learning algorithms for both single-thread and multi-thread implementations on a server. Energy usage reductions at the micro-benchmark level are substantial, while for applications obtained reductions range from 3.90% to 99.18%.

Keywords: Energy, Scientific application, Java, Micro-benchmarks, Code refactoring.

1. Introduction

Scientific computing is a field that applies Computer Science to solve scientific problems from other disciplines, such as Mathematics, Engineering, Biology, Physics and Chemistry. Scientific computing is inherently associated with large-scale computer modeling and simulation since it mainly concerns wisely using many computing resources to quickly deliver results for ever-growing problem sizes. In fact, the high popularity of this in-silico approach to research has significantly grown over the last years, which gave birth to Computational Science, a relatively new multidisciplinary

* This is an extended version of https://doi.org/10.1007/978-3-319-31232-3_69

field that uses advanced computing capabilities and notably High-Performance Computing (HPC) infrastructures to solve complex problems.

Irrespective of the computing infrastructure, research in HPC has traditionally focused on executing computations as fast as possible. Much research spanning the high-level architecture of such infrastructures including advances at the hardware level (e.g., more/faster cores for CPUs), platform level (e.g., efficient/robust middleware-level schedulers) and application level (e.g., parallel programming models) has been conducted. Nevertheless, the area has already acknowledged the importance of energy usage as well [6]. Energy consumption accounts for 15% of the operational expenditures in datacenters [17]. Furthermore, the energy consumed in datacenters in Western Europe will increase 100 TWh per year by 2020 [14], which is significant considering that for example 108 TWh was the energy consumption of Netherlands itself during 2014 according to the CIA World Factbook. This leads to huge operational costs, reduced system stability and negative ecological consequences [7].

In response, there is a wide spectrum of research efforts at the hardware level. This involves equipping processors with finer "C-states"/"P-states" and better voltage/frequency scaling techniques. Other ambitious efforts have produced the first ARM-based HPC cluster [35]. Moreover, efforts at the platform level include re-designing operating systems for energy efficiency and providing parallel middlewares to properly trade-off obtained performance and used energy for computations [1]. However, literature shows that there are few efforts focused on how HPC applications should be coded to use less energy [31, 27].

We study the energy consumed by versions of micro-benchmarks representing common programming operations found in scientific applications. To this end, we revisit a recent study [36] that has catalogued such operations but measured their implications in the context of Android programming. The experiments performed in this paper using fixed hardware show that, for the same operation, there are versions which are much more energy-efficient than others. We considered several scientific applications [12] and refactored their implementation code using the energy-efficient versions of micro-benchmarks, again obtaining energy savings. We limit the scope of our research to Java, which is useful for developing HPC applications and middlewares [41] because of its "write once, run anywhere" philosophy. This work is based on an earlier conference version published in [25], but it introduces several pertinent enhancements, namely:

1 A deeper analysis of the reasons behind the obtained differences in energy consumption for the various micro-benchmarks and their variants.
2 The use of representative scientific application kernels (SFA) as scientific test applications by basing on the well-known Phil Colella's categorization [12, 4], who identifies and delineates a set of scientific kernels which form the basis for most of the existing scientific applications. We also consider Machine Learning algorithms, the base of many real-world applications.
3 An active power versus computation time analysis of the above SFAs by considering single-core and multi-core versions of the applications.
4 Statistical significance tests to ensure results validity.

Next Section discusses related works. Section 3 explains the micro-benchmarks and details the SFAs used. Section 4 presents the experimental results. Section 5 presents the conclusions and future works.

2. Related Work

In this section, we will describe relevant efforts to increase energy efficiency in datacenters paying special attention to those focused on the latter, since our goal is reducing energy consumption via code refactorings in HPC applications.

To analyze energy consumption it is necessary to know which hardware resources consume more energy. The main part of power consumed by a server is accounted for the CPU, followed by the memory [26]. Based on this, Chen and Shi [10] present a process-level power profiling tool and a power-aware system module that eliminates energy wasted by abnormal-behavior applications for which hardware information is essential. The authors encourage the design of simple energy models to obtain real and instant measurements to control energy consumed by applications.

Other scientists analyze energy consumption of both hardware manufacture and use, and software execution [2]. For the use phase, Ardito and Morisio [2] present generic guidelines to achieve energy efficiency at four different levels: Infrastructure, Application, Operating System and Hardware. At the application level the guidelines include Design efficient UI, Use event-based programming when possible, Use low-level programming, Reduce data redundancy, Reduce QoS/scale dynamically and Use power/energy profiling tools.

Pinto, Soares-Neto and Castor [33] review works in the area of mobile programming, and they conclude that such works are focused on 6 issues to reduce energy consumption: user interface, CPU offloading, HTTP requests, software piracy, continuously running apps, and I/O operations. The authors also review efforts in the area of parallel programming, identifying 3 issues: excessive copy chains, embrace parallelism and GPU programming. However, authors do not analyze works based on servers. Besides, unlike [2] and [33], we study *concrete* energy-aware programming primitives in HPC code.

The work reported in [38] studies OO design patterns energy consumption in server applications. A new tool for measuring the power consumption and mapping between energy usage and design patterns is proposed. The authors focus on 15 creational, structural and behavioral patterns. Notable conclusions are the usage of design patterns can both increase and decrease the amount of energy used by an application and the usage of design patterns within a category impact energy usage differently.

With regards to application detailed design, Dhaka and Singh [13] study how much the correction of a wrong design affects energy consumption based on code smells, namely god class, feature envy and long method. The authors show that code smell removal permutations yield varying levels of energy consumption for the resulted software versions. It is also observed that the order in which smells are removed affects energy consumption differently. In addition, the authors propose the best sequence that generates a better design code and consumes the least energy possible.

In these lines, some works measure, control and compare energy consumption of languages, libraries, algorithms and applications. The work in [29] presents the

POWERAPI architecture which working together with power modules allows developers to calculate the power consumption of both processes and applications. With this, authors conclude that Java using the default options is quite energy-efficient in comparison to other programming languages, the energy efficiency of Pascal is at the same level as C or C++, and Perl is the most energy-consuming language. The work in [45] goes even further and analyzes execution time, memory consumption and energy consumption of 27 different programming languages over 10 different problems from the Computer Language Benchmarks Game[1]. To increase significance, the authors employ state-of-the-art compilers, virtual machines, interpreters and libraries. The main finding is that C remains as the fastest and most energy efficient language, together with compiled languages in general. In addition, Java is among the top-five most energy-efficient languages, while the least efficient ones are all interpreted.Other works evaluate common practices use or choices when developing applications. Procaccianti, Fernández and Lago [34] evaluate two practices: use of efficient queries (i.e. avoiding indexation mechanisms or unnecessary ordering operations such as SQL 'ORDER BY') and put applications to sleep to reduce CPU (and energy) utilization at the expense of increased execution time. They measure the impact using the Apache WebServer and the MySQL Server. In [27] an exhaustive evaluation of the energy consumption and performance of the NAS parallel benchmarks (NPB) is reported. The authors focus on the impact of multithreading and consider different number of threads and compilers. Authors conclude that it is difficult to balance performance and energy even for relatively simple benchmark as NBP.

Other works study the role of data structures and collections. Energy consumption of operations done on Java List, Map, and Set abstractions (e.g., insertion, iteration, random access) has been evaluated in [19]. Authors found that choosing the wrong Collections type in an application can consume 300% more energy than the most efficient collection. Second, Manotas, Pollock and Clause [24] describe an automated energy optimizer based on code-level changes. Consequently, the authors propose a framework that a) generates different versions of the same code combining all Collections instantiations, b) performs power-monitored executions of all generated versions, c) analyzes the results, and d) generates an optimized version of the original code. In the same line, jStanley [43] is a static code analyzer, implemented as a plug-in for the Eclipse IDE, which focus on reducing energy consumption by replacing Java collections for alternative, more efficient ones. The plug-in finds and quantifies method calls to collections in an application's code (maps, lists, and sets), computes normalized method calls costs, and suggests optimizations. Normalized costs are taken from a previous study from the same authors [44], where they tested the energy costs of 24 implementations of Java sets, lists and maps, considering 42 different methods in total. Interestingly, jStanley allows the user to focus on energy-driven or time-driven optimizations. Reported energy gains using real applications range from 2% to 17%.

3. Common Operations in Scientific Applications

We study eight groups of micro-benchmarks because of their recurrent use in standard and specifically scientific OO programming, namely array copying, matrix traversal,

[1] http://benchmarksgame.alioth.debian.org/

string handling, use of arithmetic operations, exception handling, object field access, object creation and use of primitive data types.

Over the years several built-in facilities were developed in diverse OO languages such as Java and C++ [32]. Then, we determine particularly energy improvement using such facilities to copy an array over implementing manually the same functionality. Additionally, matrices and related operations are important in linear algebra algorithms [28]. Regarding string manipulation, concatenation is the most important operation [11]. Concerning the fourth group, several studies have focused on optimizing arithmetic operations or involve large numbers of them [36]. Exceptions represent a widely used mechanism for elegant error handling. Method invocation was chosen since in OO programming methods must be called to use any subroutine associated with a class. In addition, we chose object creation because it involves costly memory management chores, such as garbage collection in Java or explicit object disposal in C++. Finally, the last group is the use of primitive data types versus (heavier) object-based data types.

3.1 Array Copying (AC)

Most languages include reusable libraries and built-in functionality such as data structure sorting or image manipulation. Using this support has advantages over using ad-hoc implementations since efficiency of such libraries tends to improve over time, which motivated us to compare the use of System.arraycopy method with a manual solution for the same functionality. Arrays are very important in scientific code, e.g. in mathematics arrays are used for representing polynomials.

3.2 Matrix Traversal (MT)

Matrices have many different uses such as writing problems conveniently and compactly or helping to solve problems with linear and differential equations. Additionally, in graph theory an adjacency matrix can be naturally associated to each graph where the position [i,j] indicates if vertex i is connected with vertex j.

Indeed scientific programmers use these structures quite frequently. Matrices are used to store any data type for information handling (i.e., primitive data types or objects) and are a common structure in rendering applications, where they are often used to represent and apply transformations to images. Basically, we tested micro-benchmarks where NxM matrices are traversed by rows and columns. Specifically, both micro-benchmarks involve instantiating a matrix in main memory with numeric values, using a nested loop to iterate the matrix, and accessing each cell while placing the cell value in a local variable.

Java represents n-dimensional arrays by using nested 1-dimensional arrays, which involves in principle more instantiated objects. In addition, the way this nested structure is traversed in a code might exercise the memory hierarchy differently.

3.3 String Handling (SH)

Java applications use the String class to save/read data or display messages to the user. Concatenating smaller data chunks is necessary to create bigger data chunks, thus we work with the "+" operator versus using the StringBuilder class, which exploits buffering. Despite the string concatenation operator is optimized by the compiler using the StringBuilder class, to operate using the String class and its operator "+" might yet be an inefficient practice since each concatenation with this operator implies creating a StringBuilder instance. The operator applied on n strings has $O(n^2)$ complexity, and requires memory space to maintain intermediate concatenations. We consequently expect an energy improvement using StringBuilder.

3.4 Use of Arithmetic Operations (AO)

Arithmetic operations are commonplace in scientific applications. This is illustrated for instance by data compression and mathematical applications. Also, scientific applications often need millions of calculations. Thus, the more energy-efficient the arithmetic operations are, the lower the energy consumption becomes. Since addition is one of the commonest arithmetic operation CPUs solve, we measure energy consumption of adding primitive types (int, long, float and double). Specifically, the micro-benchmark performs the successive addition into a local variable of the content of another variable whose value does not change and is set upon executing the micro-benchmark. Both variables are of type T, with $T \in \{int, long, float, double\}$. In addition, we used proper default values for the second variable (i.e. using suffixes/floating point literals) to avoid implicit upcasting/downcasting operations. Since integer operations are more efficient than floating point operations due to the greater inherent computational complexity of the later, we aim at quantifying the reduced energy consumption.

3.5 Exception Handling (EH)

Exceptions are used to manage any unexpected event in the code, while ensuring code readability. When an object is in a condition it cannot handle, it raises an exception to be captured by another object. The Java Virtual Machine (JVM) searches backward through the call stack to find methods that do can handle the exception. Sadly, exception handling is expensive and involves object creation. Then, we analyze two equivalent approaches to trigger error or exceptional situations: one using exceptions and one without these to increase energy efficiency. The tested code checks whether a numeric parameter is even and if so it always raises an exception in the inefficient version of the code, and always returned a value indicating the situation in the efficient version. In practice, the second approach implies e.g. returning an error code, an error message or an invalid value, which is a simple task for programmers. The first approach intuitively is less efficient, but the goal of the experiment is to quantify how much can be reduced by employing the second approach.

3.6 Object Field Access (OFA)

Classes comprise attributes/fields, and methods with behavior. The OO paradigm encourages information hiding, so each class should provide special public methods (accessors) used by other classes to access fields in the declaring class. However, invoking accessors also has a negative impact on performance and clearly consumes energy. For our purposes we measured the energy consumption to obtain a non-static attribute value, which in one case is performed through a method call, and in the other is performed directly, i.e., without having accessors.

3.7 Object Creation (OC)

Object creation is inherent to OO because different entities with different states coexist in memory at runtime, but this involves some computational –and hence energy– cost. However, sometimes developers can avoid creating new objects of the same class by reusing objects of that class no longer used after resetting their attributes.

We analyze the impact of object creation versus reuse on energy consumption. In other words, this means creating a new instance of an application class each time it is needed, or reusing the same instance while resetting its internal state. As the possibilities to evaluate this aspect are quite diverse because of the different classes and reset behaviors that could be implemented, we chose Lists, which are often used in applications to store data in memory and are constituting parts of other data structures. Particularly, we compare the energy consumption of creating a new list (specifically ArrayList) object and insert a String into it, versus creating an instance of ArrayList once, adding the String and using the *clear()* method to reset the list instance to its empty state.

3.8 Use of Primitive Data Types (PDT)

Past programming languages only had primitive data types (integers, booleans and strings) and procedures. Developers could define their own procedures and chain them to build larger programs based on primitives data types only, but abstract types appeared later. Java has classical primitive data types that are not classes per se, but in addition each of them has a corresponding object data type (e.g., int → Integer). We then evaluate the energy consumption using primitive data types versus using object data types. For this, we test the common behavior of accumulating several values (primitive long values) into a variable V. In one case, V is of type Long, and in another case V is defined as long.

3.9 Energy-efficient Micro-benchmarks: Test Applications

We also studied savings when refactoring real-world scientific applications based on the energy-efficient versions of the micro-benchmarks. The source code was modified considering our energy-driven optimizations only, to avoid introducing potential bias

due to unintentional inclusion of other optimizations that might also contribute to further reduce energy (e.g. removing program console output). Specifically, we refactored code by just removing all occurrences of the less-efficient micro-benchmarks to apply the most efficient ones instead, which implied for example removing all exceptions and use return values in methods, resetting the same object state rather than creating a new instance each time, using primitives data types instead of wrapper classes, and so forth.

We framed our application selection based on the Phil Colella's categorization [12, 4], who identifies a set of scientific application kernels which form the basis for most existing scientific applications. We also included Machine Learning (ML) algorithms since they are widely used in a broad range of areas, such as Bioinformatics, Natural Language Recognition and Economics. To select actual projects implementing these applications, we analyzed several sources: the Ibis/Satin parallel middleware [22], the GitHub code repository and the Weka ML library [18].

From GitHub we used JAligner[2] and gradient-descent. This later is no longer available at GitHub at the time of writing this paper, and due to licencing issues, only the binary version of gradient-descent is provided by us together with the software for reproducing our experiments. From Weka we used the Bayes Network Classifier. Lastly, another four applications were extracted from the Ibis/Satin middleware.

3.9.1 Scientific Application Kernels (SFA)

Broadly, SFAs are a set of patterns that can represent broad types of scientific applications. They are in general very CPU-intensive and use primitive data structures, such as arrays and matrices.

Phil Colella's work [12] identifies a list of seven high-level numerical methods (*dwarfs*) that represent the majority of HPC science and engineering applications, and have persisted over time. That list was enlarged in [4] to consider 6 new SFAs. To both cover some of the SFAs from [12] and [4] via applications that might benefit from as many of the micro-benchmark groups explained above as possible, we using the following concrete applications:

1. Fast Fourier Transform (FFT), which can be categorized as Spectral Methods [12]. Spectral Methods are a set of techniques to solve certain differential equations, and for that purpose they use FFT.
2. Matrix Multiplication (MMult): [4] this SFA is considered as Dense Linear Algebra one, level 3 (matrix-matrix operations). These SFAs often include access to all the elements of the data structures.
3. Knapsack (KP): This problem lays in the Backtracking and Branch & Bound category since this is a combinatorial optimization problem. Backtracking and Branch & Bound SFAs are used in Integer Linear Programming and Boolean Satisfiability as well.
4. N-Queens (NQ): This problem is one of the most characteristic type of problems found in Backtracking and Branch & Bound. It solution involves using a

[2] JAligner Web page: https://github.com/ahmedmoustafa/JAligner

modified version of Backtracking to place the queens in the different possible positions of a board.

5. Sequence Alignment (SA): This is an algorithm used to align two DNA sequences in order to analyze their similitude. To this end, Sequence Alignment algorithms usually rely on Dynamic Programming.

Table 1. Test applications. Columns are AC (Array copying), MT (Matrix traversal), SH (String handling), AO (Use of arithmetic operations), EH (Exception handling), OFA (Object field access), OC (Object creation) and PDT (Use of primitive data types)

Application	AC	MT	SH	AO	EH	OFA	OC	PDT
FFT (Fast Fourier Transform)	-	-	Yes	Yes	Yes	Yes	Yes	Yes
MMult (Matrix Multiplication)	-	Yes	-	Yes	Yes	-	Yes	Yes
KP (Knapsack)	Yes	Yes	-	Yes	Yes	-	Yes	Yes
NQ (N-Queens)	-	Yes	-	Yes	Yes	-	-	Yes
SA (Sequence Alignment)	-	Yes	-	Yes	-	Yes	Yes	Yes

Table 1 summarizes the characteristics of these applications. The first column lists the test applications, while the rest of the columns are AC (Array copying), MT (Matrix traversal), SH (String handling), AO (Use of arithmetic operations), EH (Exception handling), OFA (Object field access), OC (Object creation) and PDT (Use of primitive data types). The cells indicate whether each micro-benchmarks group was present ("Yes") or not ("-") in the various applications, and hence whether the associated energy-aware refactoring opportunities apply or not. The extent to which each application uses each micro-benchmarks group naturally varies across applications. For example, FFT instantiates more objects at runtime than the rest of the applications. Applications on the other hand do not contain many input/output operations (disk usage) that might introduce noise in the energy measurements.

FFT. It computes the discrete Fourier transform, which has an impact on different areas such as image (JPEG) and audio (MP3) processing, reduction of noise in signals, analysis of frequency of discrete signals, among others. Being x_0, x_1, ..., x_{n-1} complex numbers, directly evaluating the well-known discrete Fourier transform (DFT) formula requires $O(n^2)$ arithmetic operations. However, Gauss proposed a method that requires O(n log n) steps to evaluate it, called FFT.

The algorithm in this paper is a recursive decomposition of the FFT in simple functions until obtaining 2-element functions with k={0 or 1}. Once these simple transforms are solved, the algorithm groups them in other top level computations to be solved again until the highest recursive level is reached. Lastly, the results must be reorganized obtaining the same results as the original FFT.

Mmult. It takes as parameters two matrices (A, B) containing numbers and returns another matrix (C) which holds the result of multiplying the first two matrices. Each cell c_{ij} is the addition of the products of each element in row *i* in matrix A with the corresponding element in column *j* in matrix B.

To produce the C matrix, the application used in this paper first divides each input matrix into four quadrants. This division is recursive until the last level where there is an *nxn* matrix with *n* given as a parameter. The result at any level can be computed as

$C_{11} = A_{11} * B_{11} + A_{12} * B_{21}$; $C_{12} = A_{11} * B_{12} + A_{12} * B_{22}$; $C_{21} = A_{21} * B_{11} + A_{22} * B_{21}$; $C_{22} = A_{21} * B_{12} + A_{22} * B_{22}$. We used *n=1* to evaluate the impact of the micro-benchmarks in the most extreme case.

KP. This is an NP-complete combinatorial optimization problem whose goal is to optimize the total value that a backpack can contain. The backpack can support a default weight *W*. The backpack is filled with elements each having a value *v* and a weight *w*. The problem arises constantly in Engineering [3] and has several applications in operation management and logistics. The version used in this paper divides the initial *N* elements into two subproblems recursively for *N−1* elements, one with the lost item placed in the backpack, and the other without it. This runs recursively until the backpack is full or there are not elements left.

NQ. Implements a classic NP-hard problem where *n* queens are placed on a NxN board so that queens can be attacked considering the chess rules. The problem has been broadly used as part of more complex applications such as OS deadlock prevention and register allocation, traffic control, robot placement for maximum sensor coverage, and many others. N-Queens is also used in many other Physics, Computer Science and industrial applications [39]. The variant used in this paper searches for every possible solution, so it is very CPU intensive.

SA. Given two DNA sequences identifies the similarity regions. A sequence is represented by a string of characters, being each a *residue*. If two DNA sequences are arranged next to one another and their most similar elements juxtapose, they are aligned. There are two types of alignment methods: global and local. The former performs the alignment of all the residues of every sequence at the same time. The local approach looks into some parts of each sequence and compares them with one part of the other. This paper focuses on the Smith-Waterman [40] local alignment algorithm, which is based on dynamic programming.

3.9.2 Machine Learning Algorithms

Machine Learning (ML) involves algorithms to allow the computer to “learn”. They take as input a structured dataset, with several properties (features) to build a model able to make estimations for new data. Supervised ML algorithms are designed for datasets where each entry has associated a set of feature values and an output –usually a category. Supervised algorithms can be further divided into classification algorithms, which target discrete outputs, and regression algorithms, which target continuous outputs. Unsupervised algorithms are applied in datasets with features data but no output. Their purpose is to find relationships among the data and split it into different cohesive groups.

ML algorithms are CPU-intensive, and may take a long time to come up with a model. In addition, they are usually modeled with matrices, and lots of operations are done with those matrices. Particularly, we will study with Gradient Descent and Bayes Network Classifier. The first algorithm is the basis for many other ML algorithms and can be categorized as *Dense Linear Algebra* according to [12]. Bayes Network Classifier is a classification algorithm that uses the Bayes theorem as the basis to build the model, and is classified as *Construct Graphical Models* according [4].

Table 2. ML applications.

Application	AC	MT	SH	AO	EH	OFA	OC	PDT
Bayes	Yes	Yes	-	Yes	-	Yes	Yes	Yes
GD (Gradient Descent)	-	Yes	-	Yes	Yes	Yes	Yes	Yes

Table 2 summarizes the two ML applications employed. The first column lists the ML applications used, while the rest of the columns are AC (Array copying), MT (Matrix traversal), SH (String handling), AO (Use of arithmetic operations), EH (Exception handling), OFA (Object field access), OC (Object creation) and PDT (Use of primitive data types). Cell values are interpreted as those in Table 1.

Gradient Descent (GD). When dealing with several variables in a function, it is computationally expensive to determine its derivative to find the global minimum. Gradient Descent iteratively optimizes until convergence the search of the local minimum for a function based on the function's gradient. In fact, most ML algorithms base their calculations on this approach or on a modified version of it [8], such as Logistic Regression, Neural Networks and Deep Learning. There are basically three types of Gradient Descent: Batch, Stochastic and Mini-batch. The first one takes into consideration the whole dataset at each iteration. The second variant performs an update round for each data point of the dataset. This is usually much faster than Batch Gradient Descent and can also be used in online learning algorithms, but it may not converge to the local minimum every time. The third approach takes groups or batches of k data points. Thus, it takes the best of the two previous alternatives (fast convergence and good solution quality).

Bayes Network Classifier (Bayes). The Bayes Network Classifier is an ML supervised classification algorithm that takes advantage of the well-known Bayes theorem to classify instances in a dataset. The dataset is processed to learn the importance that each feature has in determining the category of an instance and thus classify unknown instances. Bayes Network classifiers are used in a wide range of areas, such as information retrieval, Bioinformatics, or image processing.

The commonest variant is the Naïve Bayes Classifier, which assumes that each feature is conditionally independent from all the other random features. This usually generates a high bias in the model and reduces effectiveness. Therefore, an alternative approach [15] considers the concept of Bayes Network, which depicts the dependencies between each feature in the model.

4. Experiments

We measured the individual impact of micro-benchmarks on energy consumption (Section "Micro-benchmarks Results") and their effect on the real code described earlier (Section "Test Application Results").

The JVM includes a dynamic compiler that optimizes the parts of a program that are most frequently used [5], and a *garbage collector*, periodically launched to free unused memory. These features introduce "noise" when profiling programs, especially when these programs perform fine-grained operations, like our micro-benchmarks do. Thus, we used Google's Caliper [16], a framework for running benchmarks that deals with these problems. This research considered Java 8.

The seven applications –SFAs– were also run in multi-thread mode. Given a single-thread SFA, its multi-thread counterpart was obtained by creating several instances of the SFA in a black-box fashion, one per available core in the host computer. This was done using the Executor support of Java. For the sake of uniformity, each instance was parametrized with the same parameters as the single-thread version (primitive values or object instances depending on the case). Measuring the energy consumption of the applications running in parallel would show whether there is a relationship between energy consumption and either exploiting one CPU core or multiple CPU cores.

Note that this black-box, embarrassingly-parallel scheme to run instances of a single-thread code is actually a very popular way of conducting simulation-based experiments among scientists and engineers [46]. Many of such simulations execute the same application code (e.g. a metal deformation model) in parallel with varying values for certain parameters (e.g. applied tension) resulting in different output results (e.g. did the piece broke in each case?).

With respect to quantifying energy, the PowerMeter device[3] was used. It takes 2,000 samples (voltage, amperage, active power and apparent power) per second. We plugged a host computer –4-core AMD A8-5600K APU processor (running @3600 MHz), 8 GB RAM DDR3 and Ubuntu 17.04– to the device, which was in turn plugged to the power line. The computer connects to the device via a MODBUS RS232 port. Note that this setting means that the device cannot differentiate how much how power is consumed by a given experiment and the bare system (i.e. the software which runs when the computer is idle, mainly the operating system). In consequence, the power measured in an experiment corresponds to the whole system (computer). To quantify as accurately as possible the impact of the reduced power consumption introduced by refactoring code, we aimed to reduce the consumption levels of the computer by turning off both the network card and the screen in the computer. Running an application involved several *iterations*, for the sake of decreasing statistical errors. Upon executing an iteration, we force the application to wait until the JVM is warmed up, i.e., the state at which necessary data structures, user-level threads and internal JVM threads have been initialized. We chose *iterations* = 10, which yielded deviations < 2% for all tests.

In addition, we noted that some readings from PowerMeter were invalid (i.e., apparent power was close to 2^16), so proper support was included in our experimentation software to discard such readings. Given an individual measurement log, which therefore has stored measures corresponding to the iterations of an application, only the lines having invalid apparent power values were deemed inconsistent and hence not considered upon processing the active power readings from the log. This could be done since the standard deviation of the remaining (valid) lines was, in terms of active power, below 2%, as explained above. These actions, together with the use of Caliper, allowed us to obtain correct and usable measurements.

The experimentation software (mainly bash scripts and to a lesser extent Python code), the code itself to talk to the measurement device (written in C), and the source/binary code used in the experiments are available at a GitHub repository[4].

[3] PowerMeter Web page: http://www.powermeter.com.ar/eco/
[4] https://github.com/cmateos/Experiments-ComSIS-2019

4.1 Micro-benchmarks Results

Table 3 depicts the average power consumption (in Ws) of each micro-benchmark version. Table 4 depicts the same for the Use of arithmetic operations micro-benchmark. Within each micro-benchmark least to most efficient versions are ordered from top to bottom. EnergyUsageReduction per micro-benchmark was defined as:

$$\frac{\left[\frac{\sum Ws(exec_i) - \sum Ws(improvedExec_i)}{\sum Ws(exec_i)}\right]}{\#\ of\ iterations} * 100 \quad (1)$$

where $Ws(exec_i)$ is the consumption of iteration i of the original version of a micro-benchmark, and $Ws(improvedExec_i)$ is the consumption of an individual iteration of an improved micro-benchmark version. Ws consumed by an individual iteration is the sustained active power (in Watts) as measured from the power device considering valid readings, multiplied by the time it takes to execute the iteration (in seconds). Since the power device outputs a line of data every second, the sustained active power is the average power measured during the iteration, which was possible to use as a meaningful statistical indicator since as explained low deviations were observed even discarding the invalid readings in each iteration. In the formula, we sum up all the Ws values and then divide by the number of iterations since clearly such values might be different between individual iterations.

Array copying. To compare the efficiency of System.arraycopy we used a manual implementation of the same functionality with an array of 8KB, i.e., the default internal array size in Java for buffered readers, which are extensively used for data streams. The built-in implementation reduces energy consumption by a 37.9%. These results are in line with previous studies on Java optimization [42], where using the System.arraycopy function instead of manual array copy for the entire Java I/O piped stream subsystem resulted in likewise performance gains.

Table 3. Micro-benchmarks results (*Use of arithmetic operations* not included)

Micro-benchmark	Version	Consumption (Ws)	Energy reduction (%)
Array copying	Manual array copy	102.8	
	System array copy	63.8	37.9
Matrix iteration	By-column iteration	53,776.8	
	By-row iteration	102.6	99.8
String handling	String concatenation (+)	4,456.1	
	String builder	271.7	93.9
Exception	Use Exception	14,108.6	
handling	No Exception	28.1	99.8
Object field access	Accessor-based access	9,190.0	
	Direct access	1,700.8	81.4
Object creation	On-demand creation	813.1	
	Object reuse	461.6	43.2
Use of primitive	Use of object data types	3,082.3	
data types	Use of primitive data types	2,356.2	23.5

Table 4. *Use of arithmetic operations* micro-benchmark results

Version	Consumption (Ws)	Energy reduction (%)		
		Versus double	Versus float	Versus long
Add constant to double	5,152.5	-	-	-
Add constant to float	5,089.3	1.2	-	-
Add constant to long	3,643.5	29.2	28.4	-
Add constant to int	838.8	83.7	83.5	76.9

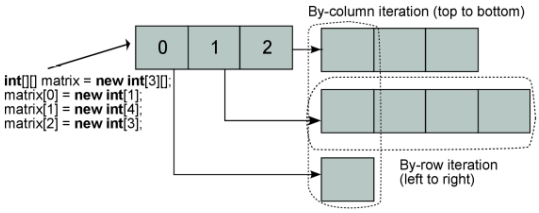


Fig 1. Two-dimensional array representation and traversing in Java

At the JVM level, using manual array copy implies copying array elements one by one, whereas invoking System.arraycopy delegates the copy to a native method. A native method be implemented differently by each JVM runtime and can be optimized in several ways that are not a possibility for Java developers. For example, the copy of the array can be done with a single *memcpy/memmove* low-level primitive from a native method, instead of *n* distinct copy operations.

Matrix traversal. This paper uses NxM matrix structures and compares traverse by rows versus traverse by columns. Specifically, a matrix of 1024x1024 was used to run tests. A key advantage of these micro-benchmarks is the simplicity of changing the traverse mode in an existing code. The results show an improvement (energy reduction) of 99.8% using the traverse by row version.

Java represents two-dimensional matrices via an array, where each cell points to another object array (Fig. 1). Overall, when a matrix is traversed by row, all the cells of arr [0] are traversed first, continuing with arr[1] and so on. When reading arr[0][0], the CPU caches the cells that are close by (arr[0][0] to arr[0][n] and may cache some cells from the next row). When the matrix is traversed by row, the next cell (arr[0][1]) is likely cached, which is faster than fetching the cell from main memory. But, when traversing by column, some of the next cell accesses (arr[1][0], arr[2][0], ..., arr[n][0]) are likely to cause a cache miss.

String handling. Table 3 shows that using the class StringBuilder directly instead of the "+" operator yields a very good improvement (1,000 concatenations were used). String literals in Java are instances of String, which are immutable meaning that their characters cannot be changed after created. Using the "+" operator involves the creation of a StringBuilder object that maintains a single internal *mutable* array of characters. Besides, the method using "+" also instantiates the StringBuilder class to handle concatenation, but performs four method calls whereas the efficient version performs three method calls.

Use of arithmetic operations. This micro-benchmark group, whose results are shown in Table 4, involved adding a constant value *c* to a numerical variable declared by varying their data type. Specifically, we resolved $X + c$ using float, double, int and long variables and constants. As a result, using the float, long and int data types yields a reduction of 1.2%, 29.2% and 83.7% respectively over relying on the double data type.

Then, double and long data types consume more energy than float and int data types, respectively, because the former provide greater accuracy and larger range of values. This means more bits to represent values and therefore more processing time. In practice, programmers should of course to keep accuracy and precision as low as possible for numerical data types in order to reduce energy consumption while not compromising the semantics of the whole application.

Exception handling. The results in Table 3 confirm that energy can be saved by avoiding exceptions. The creation of objects and the limited optimizations to the exception mechanism made by the JVM, produce higher energy consumption. To ensure minimum consumption, exceptions must be reserved only for error situations where cannot be dealt with other mechanism, for example when using third-party libraries within the application code that are designed to communicate error situations via exceptions.

An operation that includes an exception throwing executes the same lines as the same operation without exceptions but it also adds an object creation and new JVM instructions processing. Developers should define error statuses instead of using exceptions whenever possible to deal with abnormal execution flows.

Object field access. Directly reading a frequently-accessed class field yields an improvement (81.4%) because the accessor method invocation is avoided. Despite this, programmers must determine to what extent it is valuable to violate object encapsulation to favour energy efficiency. However, there are common cases in which encapsulation is not affected and energy can be reduced, e.g., accessing a class field directly from the same class or inner classes.

Object creation. By reusing objects an energy reduction of 43.2% was obtained. At the JVM level, the cost to create a new object is usually higher than the cost necessary to reset an already created object. In particular, reusing an instance of ArrayList only involves invoking its *clear()* method. This latter is efficiently implemented by just zeroing the head pointer in the internal array.

This result means developers concerned with minimizing energy consumption should not create objects arbitrarily in the code but reuse instances whenever convenient. However, energy reductions may vary depending on the objects to create: those with costly "reset" methods could outweight the benefit. In these cases, a deeper pros-cons analysis is necessary. Indeed, when running the same micro-benchmark by using Vector and LinkedList, which together with ArrayList are three of the most popular linear data structures in Java, the gain of the performed refactoring for Vector is very close to that of using ArrayList, but the refactoring increments energy usage by 1% when using LinkedList.

Use of primitive data types. The use of primitive data types yielded an energy saving of up to 23.5%. If primitive data types are used, the creation of new objects by the JVM to maintain object types is avoided. Indeed, in the previous micro-benchmark, it was shown that object creation leads to higher energy consumption. In addition, extra energy is saved since autoboxing and unboxing operations are not needed when using primitive

types. Autoboxing/unboxing is the conversion by the JVM from/to primitive types to their corresponding object type.

4.2 Test Applications Results

Table 5 and Fig. 2 show the resulting energy consumption, where the reductions in % of the refactored versions according to our micro-benchmarks with respect to the original ones have been quantified as explained earlier. Multi-thread code used the 4 cores available. Next we discuss in detail the obtained results.

FFT (Fast Fourier Transform). The main refactoring on this application was the elimination of immutable classes. This was possible through the modification of a class named Complex, which was immutable in the original test application. In the new version, Complex class instances can change the values of their attributes without creating a large number of immutable instances of such class. Also, the attributes precision of the Complex class (i.e. its real and imaginary part) was decreased from double to float without altering the FFT algorithm itself.

It is worth noting that by changing from double to float we are potentially losing precision. In Java, the double data type is 64-bit wide, with precision of up to 15 to 16 decimal points. The float data type is 32-bit wide, with precision of up to 6 to 7 decimal points. All in all, whether losing precision is problematic will depend on the application exploiting the FFT algorithm. For example, 32-bit precision suffices many audio processing related tasks.

Mmult (Matrix Multiplication). The main aspect to avoid in this test application was object creation. However, in this test application the instantiation of different classes (matrices) is performed at the beginning of the code. The matrix structure was redesigned decreasing the number of object creations: not using a recursive structure has the advantage of requiring fewer objects in memory.

KP (Knapsack). In this test application we reduced the number of objects in memory by a half. In the original version instances of the class OrcaRandom and Knapsack class were created, while in the refactored version only instances of Knapsack were created, which included the behavior of OrcaRandom.

NQ (N-Queens). This application is algorithmically rather simple. There is only one class which implements the algorithm itself, so the main refactoring for this particular case was to change the non-primitive data types and to avoid some object creation in very specific cases.

Table 5. Application results. From top to bottom, applications are listed in the order of Section "Energy-efficient Micro-benchmarks: Test Applications"

App.	Version	Consumption (Ws) / Time (s)		Energy usage reduction %	
		Single-thread	Multi-thread	Single-thread	Multi-thread
FFT	Original	1,784.76/ 27.9	947.57 / 8.1		
	Refactored	1,714.99 / 26.4	813.14 / 7	3.90	14.19
MMult	Original	34,315.15 / 496.2	22,692.00 / 183		
	Refactored	13,123.99 / 185.7	8,261.35 / 66	61.75	63.59
KP	Original	5,181.22 / 71.3	4,320.79 / 36		
	Refactored	104.94 / 1.5	103.41 / 1	97.97	97.61

NQ	Original	55,753.39 / 854	34,394.70 / 300		
	Refactored	40,315.14 / 605	22,374.64 / 189.5	27.69	34.95
SA	Original	40,813.78 / 613.6	61,832.88 / 680.7		
	Refactored	906.92 / 13.1	508.52 / 4.3	97.77	99.18
GD	Original	6,361.87 / 94	3,630.05 / 29.1		
	Refactored	3,131.95 / 44.8	1,920.94 / 15.5	50.76	47.08
Bayes	Original	14,566.14 / 114.1	1,1599.18 / 110.4		
	Refactored	11,733.39 / 88.8	1,0415.26 / 8.1	19.44	10.20

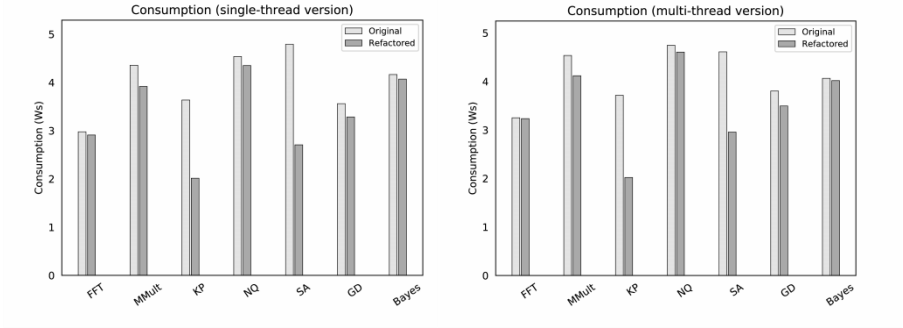


Fig 2. Consumptions for single-thread (left) and multi-thread (right) modes. Bars are log10-scaled

SA (Sequence Alignment). In the original code there is a recurrently-used class (Matrix), which is a two-dimension array of instances of the Float object type. So the most important refactoring was to use primitive data types. There were also modifications in the main class to avoid new object creations and method invocations. Note that the unrefactored multi-thread version consumed much more energy than its single-thread counterpart. As mentioned earlier, we produced multi-thread versions of applications by cloning the original application and feeding each clone with the same parameter values or instances, depending on the case. For SA, this particularly meant passing on the same object instances (two Sequence objects, representing human and mouse protein sequences), which in turn led to high memory contention among threads. However, we aimed at leaving the application code "as is" prior to refactor them and using the same multi-thread scheme for all applications, without introducing solutions to mitigate this contention. In fact, avoiding object data types and reducing object creations decreased memory usage in the refactored single-thread version.

GD (Gradient Descent). GD is a machine learning algorithm that basically learns (approximates) a multi-variable function using training data. The implementation of GD used is based on two matrices: an NxM matrix with N the number of variables and M the training set size, and another Mx1 matrix with the values of the training set. The original version of these matrixes were implemented using a Matrix class with a Collection with non-primitive data types (Double). The applied refactoring was to replace this collection with arrays of primitive data types. Thus, two further optimizations were also applied in consequence to create the optimized code. Firstly, there are less objects since one Matrix instance itself is an object.

Second, elements can be read by directly indexing an array position, i.e. without accessors. Note that this change is possible since the amount of elements in the matrices is known a priori, thus an accessor is not needed. This is possible since machine learning algorithms are usually trained with data with dimensions and sample numbers known in advance.

Bayes (Bayes Network Classifier). The implementation maps each entry of the dataset into Instance objects. Each of these objects are then processed to train the classifier. The whole set of instances (dataset) are kept in another class called Instances, which provides the methods to get or put information into it and is mainly composed by a List. In addition, similar to GD, it is possible to know the size of the dataset a priori. Thus, the refactoring applied was again eliminating the List and using an array instead.

4.2.1 Results Summary

Energy spent by an application version is computed based on active power (Watts) and runtime (seconds). For each triple T=<app, v, th>, app ∈ {FFT, MMult, KP, NQ, SA, GD, Bayes}, v ∈ {original, refactored} and th ∈ {single-thread, multi-thread}, we obtain two lists, LP and LT. LP has the active power samples from *i* iterations, and LT contains *i* elapsed times in seconds. Since our power device outputs a line of raw measurement data every one second, the size of LP is $\sum$(LT_j).

To illustrate the amount of samples in the lists, please refer to Table 5. The triple $T_{GD,o,s}$=<GD, 'original', single-thread> took 94 seconds to execute in average. LP($T_{GD,o,s}$) will then have approximately 94*10=940 samples (recall we used *i*=10 in all experiments). On the other hand, the triple $T_{GD,r,s}$=<GD, 'refactored', single-thread> took 44.8 seconds to execute in average, so LP($T_{GD,r,s}$) will have around 440 samples. Lastly, both LT($T_{GD,o,s}$) and LT($T_{GD,r,s}$) will have 10 elements, one per iteration.

We studied the source of energy reductions by performing statistical tests given T_1=<app, 'original', th> and T_2 =<app, 'refactored', th>. This means determining whether there are statistically significant differences between samples of LP(T_1) versus that of LP(T_2), and samples of LT(T_1) versus that of LT(T_2).

For energy samples, we took the active power samples lists LP(T_1) and LP(T_2) and since the lists might differ in length we run the two-tailed Mann-Whitney-Wilcoxon for unpaired data. This difference in length stems from the fact that $\sum$($LT(T_1)_j$ is usually different than $\sum$($LT(T_2)_j$, and hence the sizes of LP(T_1) and LP(T_2) also differ. For instance, the size of LP($T_{GD,o,s}$) and LP($T_{GD,r,s}$) is 940 and 440, respectively.

For elapsed times, and since the lists LT(T_1) and LT(T_2) have the same length and samples differ from each other in that a *treatment* (refactoring) is applied, we used the two-tailed Wilcoxon test for paired/matched data. This resembles the kind of test often applied on the same subject –in our case application- before and after a treatment has been applied. This is, before the treatment is applied, the application code is the original one, while after the treatment is applied, the code has been refactored. Note that each element in LT(T_1) and LT(T_2) are sampled independently, but for the sake of the statistical test they are matched, which means that the Wilcoxon test uses as input a single list with the element-wise difference of both lists.

Table 6 shows the test outcomes. Since refactored code (T_2) tended to demand more active power but less time to run than original code (T_1), we in fact tested the

significance of active power increment and elapsed time decrement of the refactored code over the original code.

Table 6. Active power and elapsed time differences: Statistical significance test outcomes (Y = Yes, N = No)

App.	Original vs refactored round	Active power decrement		Elapsed time decrement	
		At 0.01?	At 0.05?	At 0.01?	At 0.05?
FFT	Single-thread / Multi-thread	Y / N	Y / N	Y / Y	Y / Y
MMult	Single-thread / Multi-thread	Y / N	Y / Y	Y / Y	Y / Y
KP	Single-thread / Multi-thread	N / Y	N / Y	Y / Y	Y / Y
NQ	Single-thread / Multi-thread	Y / Y	Y / Y	Y / Y	Y / Y
SA	Single-thread / Multi-thread	Y / Y	Y / Y	Y / Y	Y / Y
GD	Single-thread / Multi-thread	Y / N	Y / N	Y / Y	Y / Y
Bayes	Single-thread / Multi-thread	Y / Y	Y / Y	Y / Y	Y / Y

Table 5 shows that, considering single-thread code runs, the refactored versions demanded more active power than the original versions (2-4%). The exception to this is KP, whose refactored version had 3.72% less active power. For multi-thread code, this overall trend does not hold and in fact refactored versions introduced average active power reductions compared to original code in four cases, i.e., 0.70% (FFT), 13.83% (KP), 23.18% (SA) and 0.65% (GD), which are statistically significant at the 0.01 and 0.05 confidence levels.

Another observation is that multi-thread code used more active power (between 90.83 Watts and 125.17 Watts) than single-thread code (between 63.36 Watts and 72.66 Watts). Since *Energy=ActivePower* * *RunTime*, these results show that the studied micro-benchmarks do not reduce *Energy* as a side product of *Runtime* only, but also *ActivePower* is altered.

Table 6 shows that all significant tests regarding elapsed time confirm that refactored code run faster than original code. Let us measure such improvements using the well-known speedup metric, which is the ratio between the time it takes to run an unoptimized code versus the time to run its optimized counterpart, i.e. original times over refactored times in our case. Speedups values ranged from [1.05-47.53] (single-thread) and [1.15-158.30] (multi-thread). Overall, we obtained per-iteration absolute average energy savings of 69 Ws to 39900 Ws (single-thread) and in the range of 134 Ws to 61300 Ws (multi-thread). Even when multi-thread refactored code naturally consumes more Active Power than single-thread refactored code, in the former case each core runs a refactored –and hence rather faster– version of the original code. Again, since *Energy=ActivePower* * *RunTime* the multiplicative, beneficial effect on energy consumption of using many threads can be also appreciated.

To put these savings in context, virtualization technologies –particularly Xen and KVM– and container technologies –particularly LXC and Docker– consume between 126 and 128 Ws to run eight simultaneous idle virtual guests [30]. Likewise, the energy to send 27 MB of data via TCP in metropolitan-area networks where round-trip time is up to 50 milliseconds ranges from 921 to 43000 Ws [21]. Lastly, 30000 Ws is the energy necessary to execute Kmeans clustering algorithm from the benchmark in [9] by splitting the work to do under a 50-50 scheme between a CPU and an Nvidia GeForce 8800 GTX GPU [23].

To conclude our analysis, we should also mention that the potential energy savings in an application is only an angle from which to evaluate whether it is convenient to refactor the application code or not regarding some micro-benchmarks. This way, another important angle is *analysis scope*, which refers to the quantity of code units that users have to analyze to determine where to apply refactorings without affecting the application functionality, and hence it is a qualitative measure of refactoring difficulty. This analysis might involve looking only the sections of the code where the refactoring opportunities appear, or additionally more elements like methods that call those sections or other classes. The analysis scope can be at the Statement, Method or Application levels. The Statement level particularly requires less effort from the user. For example, when refactoring for the OFA micro-benchmark, users have to change all Getter method calls by direct accesses to involved attributes (Statement level). For the MT micro-benchmark, changing the traverse orientation is a trivial task in terms of code, but it is not a trivial task at the time of analyzing the semantic of the traverse. This involves looking the method implementing the algorithm where the traverse is performed (Method level). For example, the traverse in a matrix multiplication code cannot be changed. However, after an analysis, developers could transpose the matrices and, then, change the traverse. Finally, refactoring for the AO micro-benchmark clearly implies to analyze the feasibility of reducing data types precision at the Application level.

Table 7 summarizes the micro-benchmarks based on these two angles. We have considered a qualitative indication of the energy savings that can be obtained from each micro-benchmark. In practice, this represents a prioritization for users willing to exploit our micro-benchmarks, since those yielding the best energy savings and being the most easy to apply in the code should be tackled first (e.g. OFA, EH, MT and PDT).

Table 7. Studied micro-benchmarks: energy savings and analysis scope difficulty

Micro-benchmark	Energy savings	Application scope
Array copying (AC)	Good	Application
Matrix iteration (MT)	Excellent	Method
String handling (SH)	Excellent	Application
Use of arithmetic operations (AO)	Very low-very good	Application
Exception handling (EH)	Excellent	Method
Object field access (OFA)	Very good	Statement
Object creation (OC)	Good	Application
Use of primitive data types (PDT)	Good	Statement

5. Conclusions

We have empirically assessed the energy impact of energy-friendly versions of common primitives in Java scientific code. We also show that refactoring code driven by such energy-friendly versions yield energy gains both for single-thread and multi-thread refactored applications. This gives Java scientific developers hints to build energy-efficient software for servers, which complements energy-aware approaches already proposed at the platform and hardware levels.

It is worth noting that our research benefits end user scientific applications, i.e. software whose primary purpose is not to be heavily reused (as opposed to software

libraries). In practice, refactoring an application would essentially mean modifying the original code and then properly testing the refactored code to avoid introducing bugs. However, modifying code that is aimed at being reused from other applications requires a wider view upon refactoring code to avoid breaking clients.

Consequently, if we analyze the potential impact of micro-benchmarks driven refactoring in software that is aimed at being reused, they can be grouped into those that are harmless and those that might break the software. In the former group is Array copying, Matrix iteration and String handling. Refactoring based on these micro-benchmarks means changing the way certain tasks are implemented, but software design is not broken.

Contrarily, the micro-benchmarks in the second group, i.e. the rest, might break the software design. In many cases, the library interface is affected thus breaking clients (Exception handling, Object field access, Use of primitive data types), internal object states might be violated or made inconsistent (Object creation) or what the client expects from the library might be semantically altered (Use of arithmetic operations). This does not mean our micro-benchmarks cannot be applied in libraries as well, since they would be applicable in libraries where a clear, defined separation between interface (API) and implementation exists. In this way, refactorings could be applied in principle within the boundaries of the library implementation while ensuring that the API is left untouched (both syntactically and semantically).

Finally, future work will investigate how to automatically preprocess existing code to exploit our findings. For some micro-benchmarks (e.g., object field access) this is trivial but for others (e.g., reusing objects) modification/recognition is highly challenging. We are also exploiting these ideas for mobile device programming. Preliminary works studied the rate at which micro-benchmarks versions deplete batteries [36] and the trade-off between code smell-free OO designs versus the inherent energy costs [37] in Java-based Android applications. The motivation of these works is that mobile devices can act as resource providers in edge environments to run scientific applications [20], so coding energy-aware tasks becomes crucial. In addition, we will test other common situations not covered by the micro-benchmarks code utilized in this paper. For example, these include other arithmetic operations (AO micro-benchmark), checking if a method return value is correct as opposed to having an exception (EH micro-benchmark), accessing static versus non-static object attributes (OFA micro-benchmark) and exclusively using wrapper classes in an application since boxing is avoided (PDT micro-benchmark).

Acknowledgements. We thank the anonymous reviewers for their comments to improve the paper. We acknowledge the financial support by ANPCyT through grant PICT no. PICT-2012-0045 and CONICET through grant PIP no. 11220170100490CO.

References

1. S. K. Abd, S. Al-Haddad, F. Hashim, A. B. Abdullah, S. Yussof, An effective approach for managing power consumption in cloud computing infrastructure, Journal of Computational Science 21 (2017) 349–360.

2. L. Ardito, M. Morisio, Green it available data and guidelines for reducing energy consumption in it systems, Sustainable Computing: Informatics and Systems 4 (1) (2014) 24–32.
3. J. Zhang, Comparative study of several intelligent algorithms for knapsack problem, Procedia Environmental Sciences 11 (2011) 163–168.
4. K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, et al., The landscape of parallel computing research: A view from berkeley, Tech. rep., University of California (2006).
5. A. Barisone, F. Bellotti, R. Berta, A. De Gloria, Jsbricks: a suite of microbenchmarks for the evaluation of java as a scientific execution environment, Future Generation Computer Systems 18 (2001) 293–306.
6. R. Basmadjian, P. Bouvry, G. Da Costa, L. Gyarmati, D. Kliazovich, S. Lafond, L. Lefevre, H. De, J.-M. P. Meer, R. Pries, J. Torres, T. Trinh, S. Khan, Green data centers, Large-Scale Distributed Systems and Energy Efficiency: A Holistic View (2015) 159–196.
7. J. Brożyna, G. Mentel, B. Szetela, Renevable energy and economic development in the european union, Acta Polytechnica Hungarica 14 (7) 11-34.
8. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: 22nd International Conference on Machine learning, ACM, 2005.
9. S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, K. Skadron, Rodinia: A benchmark suite for heterogeneous computing, in: IEEE International Symposium on Workload Characterization, IEEE, 2009.
10. H. Chen, Y. Li, W. Shi, Fine-grained power management using process-level profiling, Sustainable Computing: Informatics and Systems 2 (1) (2012) 33–42.
11. A. S. Christensen, A. Moller, M. I. Schwartzbach, Precise analysis of string expressions, in: 10th International Static Analysis Symposium, 2003.
12. P. Colella, Defining software requirements for scientific computing, Tech. rep., DARPA's High Productivity Computing Systems (HPCS) (2004).
13. G. Dhaka, P. Singh, An empirical investigation into code smell elimination sequences for energy efficient software, in: 23rd Asia-Pacific Software Engineering Conference, 2016.
14. European Commission, Code of conduct on data centres energy efficiency, Tech. rep., Institute for Energy, Renewable Energies Unit, v2.0 (2009).
15. N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine learning 29 (2-3) (1997) 131–163.
16. Google, Caliper, https://github.com/google/caliper/wiki/ProjectHome.
17. A. Greenberg, J. Hamilton, D. A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, ACM SIGCOMM Computer Communication Review 39 (1) (2008) 68–73.
18. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: an update, ACM SIGKDD explorations newsletter 11 (1) (2009) 10–18.
19. S. Hasan, Z. King, M. Hafiz, M. Sayagh, B. Adams, A. Hindle, Energy profiles of java collections classes, in: 38th International Conference on Software Engineering, 2016.
20. M. Hirsch, J. M. Rodriguez, A. Zunino, C. Mateos, Battery-aware centralized schedulers for cpu-bound jobs in mobile grids, Pervasive and Mobile Computing 29 (2016) 73–94.
21. M. Usman, D. Kliazovich, F. Granelli, P. Bouvry, P. Castoldi, Energy efficiency of tcp: An analytical model and its application to reduce energy consumption of the most diffused transport protocol, International Journal of Communication Systems 30 (1).
22. R. V. van Nieuwpoort, G. Wrzesińska, C. J. Jacobs, H. E. Bal, Satin: A high-level and efficient grid programming model, ACM Transactions on Programming Language and Systems 32 (3) (2010) 1–39.
23. K. Ma, Y. Bai, X. Wang, W. Chen, X. Li, Energy conservation for gpu–cpu architectures with dynamic workload division and frequency scaling, Sustainable Computing: Informatics and Systems 12 (2016) 21–33.

24. I. Manotas, L. Pollock, J. Clause, Seeds: A software engineer's energy-optimization decision support framework, in: 36th International Conference on Software Engineering, ACM, 2014.
25. C. Mateos, A. Rodriguez, M. Longo, A. Zunino, Energy implications of common operations in resource-intensive java-based scientific applications, in: New Advances in Information Systems and Technologies, Springer, 2016, pp. 739–748.
26. L. Minas, B. Ellison, Energy Efficiency for Information Technology: How to Reduce Power Consumption in Servers and Data Centers, Intel Press, 2009.
27. A. E. Trefethen, J. Thiyagalingam, Energy-aware software: Challenges, opportunities and strategies, Journal of Computational Science 4 (6) (2013) 444–449.
28. A. Nicolaos, K. Vasileios, A. George, M. Harris, K. Angeliki, G. Costas, A data locality methodology for matrix-matrix multiplication algorithm, Journal of Supercomputing 59 (2012) 830–851.
29. A. Noureddine, A. Bourdon, R. Rouvoy, L. Seinturier, A preliminary study of the impact of software engineering on greenit, in: 1st International Workshop on Green and Sustainable Software, 2012.
30. R. Morabito, Power Consumption of Virtualization Technologies: An Empirical Investigation, in: IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), 2015.
31. A.-C. Orgerie, M. D. d. Assuncao, L. Lefevre, A survey on techniques for improving the energy efficiency of large-scale distributed systems, ACM Computing Surveys (CSUR) 46 (4) (2014) 47.
32. S. Papadimitriou, K. Terzidis, S. Mavroudi, S. Likothanassis, Exploiting java scientific libraries with the scala language within the scalalab environment, IET Software 5 (2011) 543–551.
33. G. Pinto, F. Soares-Neto, F. Castor, Refactoring for energy efficiency: A reflection on the state of the art, in: 4th International Workshop on Green and Sustainable Software, GREENS '15, IEEE Press, 2015.
34. G. Procaccianti, H. Fernández, P. Lago, Empirical evaluation of two best practices for energy-efficient software development, Journal of Systems and Software 117 (2016) 185–198.
35. N. Rajovic, P. M. Carpenter, I. Gelado, N. Puzovic, A. Ramirez, M. Valero, Supercomputing with com modity cpus: Are mobile socs ready for hpc? International Conference on High Performance Computing, Networking, Storage and Analysis, ACM, 2013.
36. A. Rodriguez, C. Mateos, A. Zunino, Improving scientific application execution on android mobile devices via code refactorings, Software: Practice and Experience 47 (5) (2017) 763–796.
37. A. Rodriguez, C. Mateos, A. Zunino, M. Longo, An analysis of the effects of bad smell-driven refactorings in mobile applications on battery usage, in: Modern Software Engineering Methodologies for Mobile and Cloud Environments, IGI Global, 2016, pp. 155–175.
38. C. Sahin, F. Cayci, I. L. M. Gutiérrez, J. Clause, F. Kiamilev, L. Pollock, K. Winbladh, Initial explorations on design pattern energy usage, in: 1st International Workshop on Green and Sustainable Software, 2012.
39. P. San Segundo, New decision rules for exact search in n-queens, Journal of Global Optimization 51 (3) (2011) 497–514.
40. T. F. Smith, M. S. Waterman, Identification of common molecular subsequences, Journal of Molecular Biology 147 (1) (1981) 195–197.
41. G. L. Taboada, S. Ramos, R. R. Exposito, J. Tourino, R. Doallo, Java in the high performance computing arena: Research, practice and experience, Science of Computer Programming 78 (5) (2013) 425 – 444.
42. J. Zhang, J. Lee, P. K. McKinley, Optimizing the java piped i/o stream library for performance, in: International Workshop on Languages and Compilers for Parallel Computing, Springer, 2002.

43. R. Pereira, P. Simao, J. Cunha, J. Saraiva, jStanley: Placing a Green Thumb on Java Collections. 33rd ACM/IEEE International Conference on Automated Software Engineering, ACM, 2018.
44. R. Pereira, M. Couto, J. Saraiva, J. Cunha, J. Fernandes, The Influence of the Java Collection Framework on Overall Energy Consumption. 5th International Workshop on Green and Sustainable Software, ACM, 2016.
45. R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. Fernandes, J. Saraiva, Energy Efficiency Across Programming Languages: How Do Energy, Time, and Memory Relate? 10th ACM SIGPLAN International Conference on Software Language Engineering, ACM, 2017.
46. D. Monge, E. Pacini, C. Mateos, C. García Garino. Meta-heuristic based Autoscaling of Cloud-based Parameter Sweep Experiments with Unreliable Virtual Machines Instances. Computers & Electrical Engineering - Special Issue on 7th special section on Cloud Computing 69 (2018) 364-377.

Mathias Longo holds a BSc. in Systems Engineering from the UNICEN, and he is currently pursuing an MSc. in Data Science at the University of Southern California.

Ana Rodriguez holds a BSc.in Systems Engineering from the UNICEN, and a Ph.D. in Computer Science from the UNICEN (2018), working under the supervision of Alejandro Zunino and Cristian Mateos.

Cristian Mateos holds an MSc. and a Ph.D. in Computer Science from the UNICEN. He is an adjunct professor at the UNICEN and researcher at the CONICET. He is interested in parallel and distributed programming, middlewares, and mobile/service-oriented computing.

Alejandro Zunino holds an MSc. and a Ph.D. in Computer Science from UNICEN. He is an adjunct professor at the UNICEN and researcher at the CONICET. His research areas include grid computing, service-oriented computing, Semantic Web services, and computer security.

Received: June 08, 2018; Accepted: June 01, 2019

Computer Science and Information Systems 16(2):565–595 https://doi.org/10.2298/CSIS181001010C

Outlier Detection in Graphs: A Study on the Impact of Multiple Graph Models

Guilherme Oliveira Campos[1,2], Edré Moreira[1], Wagner Meira Jr.[1], and Arthur Zimek[2]

[1] Federal University of Minas Gerais
Belo Horizonte, Minas Gerais, Brazil
{gocampos,edre,meira}@dcc.ufmg.br
[2] University of Southern Denmark
Odense, Denmark
zimek@imada.sdu.dk

Abstract. Several previous works proposed techniques to detect outliers in graph data. Usually, some complex dataset is modeled as a graph and a technique for detecting outliers in graphs is applied. The impact of the graph model on the outlier detection capabilities of any method has been ignored. Here we assess the impact of the graph model on the outlier detection performance and the gains that may be achieved by using multiple graph models and combining the results obtained by these models. We show that assessing the similarity between graphs may be a guidance to determine effective combinations, as less similar graphs are complementary with respect to outlier information they provide and lead to better outlier detection.

Keywords: outlier detection, multiple graph models, ensemble.

1. Introduction

Outlier detection is a challenging problem, since the concept of outlier is problem-dependent and it is hard to capture the relevant dimensions in a single metric. The inherent subjectivity related to this task just intensifies its degree of difficulty. The increasing complexity of the datasets as well as the fact that we are deriving new datasets through the integration of existing ones is creating even more complex datasets. Consequently, methods have been specialized in various ways, e.g., for high-dimensional data [45,18,26], for sequence data [8], or for spatial data [38]. Graphs, which are able to express a variety of rich data relationships [2], reinforce the trend towards more complex concepts of outliers.

Ensemble techniques have been used in outlier detection (including outliers in graphs [32]). Even though ensemble techniques are not yet well studied nor established for outlier detection in graphs, this powerful approach becomes more and more frequent on relational datasets, mainly because of the benefits of combining individual outlier detection results, tackling the problem from multiple perspectives [42]. The focus of existing techniques is on the design of diverse ensemble members and on the combination strategies to put individual results together in an ensemble, but they are always relying on the information available in a given graph model of some target data. However, different kinds of outliers may be easier or harder to detect in different graph models derived from the same original raw data.

Normally, the graph representations or models found in the literature to represent real-world relations were created from decisions made by the user, defining which are the

nodes and which are the edges. Complex web sites or databases rich in information such as Facebook, DBLP, and Twitter, among others, can be modeled as graphs in numerous possible ways. For example, Facebook user data could be represented as a graph where each node represents a person and each edge represents the existence of a friendship connection between nodes. But we could also connect people that have common interests, the same workplace, university, etc. Or we could be interested in modeling events as nodes and connect two events with an edge if the same people attended both. Thus, these graph models contain a bias built in by the user when generating such graphs. Obviously this bias can be of great value to some users but counter-productive to other users, depending on the task that will be performed in such graphs. Users may create the graphs according to the properties that they define necessary to facilitate the task of extracting information in such graphs. However, the choice of such a bias, which is often overlooked, is striking in the final outcome of the task that will be executed on the graphs.

Generating multiple graph models for some raw data and combining results obtained on those different models is a strategy to tackle more complex and diverse outliers. However, the generation of a graph for some given data is problem dependent. Although the model dimensions are usually intuitive and the analyst is able to enumerate them, it is hard to assess which dimensions effectively improve the outlier detection. We, therefore, make our assessment using different graph models, following different intuitions about which data aspects we want to model, but also diversified by some random parameters. Note that we are not proposing a way to automatically generate multiple graph models given a database. One could debate whether it is even possible as just the data analyst may know different dimensions of a particular graph. We assess the similarity of graphs as a possible guiding principle in assembling ensembles in the absence of ground truth, i.e., in the unsupervised learning task of outlier detection.

Let us summarize in the following list the contributions of this work:

1. We devise a methodology for the quantitative assessment of the gains of multiple graph models for some given database.
2. We explore two similarity-based strategies for selecting ensemble members from multiple graph models.
3. We present a quantitative experimental evaluation of outlier detection ensembles based on multiple graph models using synthetic data, including comparison to single graph models.
4. We evaluate the gains of employing multiple graph models using two outlier detection algorithms and four combination techniques.
5. We perform three case studies on data derived from DBLP, from Citation Network, and from Facebook and provide quantitative and qualitative insights regarding multiple graph models.
6. We also present support to advise the practical data analyst to employ multiple graph models with a preference for more diverse graphs over just more graphs.

This paper is an extended version of our previous conference paper [6]. The main extensions are the use of an additional similarity measure (DeltaCon [17]), an additional base outlier detection method (Radar [22]), additional combination techniques (mean, max and borda), and an additional dataset (Citation Network).

The remainder of this study is organized as follows. We discuss related work in Section 2. We describe our methodology to characterize the gains of multiple graph models

and to compose an ensemble based on a set of multiple graph models in Section 3 and its implementation in case studies in Section 4. We analyze and discuss the results in Section 5. We conclude the paper in Section 6.

2. Related Work

To the best of our knowledge, there is no previous work that analyzes the impact of using multiple graph models on the quality of outlier detection. However, Rotabi et al. [34] use multiple overlapping networks to solve the strong tie detection task. They combine information provided by two different graphs (a dense and a sparse graph) from the same dataset (e.g., Twitter) to predict strong ties. They made experiments with one dense graph and four sparse graphs from their Twitter dataset: mutual follow, phone book, email address book, and direct message, respectively. To generate multiple graph models and to combine information extracted from them to improve the prediction is related to our approach. However, the tasks of outlier detection and of building ensembles for outlier detection are different and come with different challenges.

Two particular areas of outlier detection relate to our work: methods to detect outliers in (single) graphs and methods to combine outlier results (ensembles). In the following subsections we survey some methods for outlier detection in graphs, highlighting the principles of the method employed in our experiments, and sketch some relevant research in ensemble methods for outlier detection.

2.1. Outlier Detection in Static Graphs

Various methods have been proposed to detect outliers in static graphs [2]. On plain graphs, the outliers may be identified based on structural behavior [1,13] or community behavior [7,40,41]. On attributed graphs, outliers may also be identified by structural behavior, looking for unusual substructures [29,23]. We focus on detecting community (or contextual) outliers on static attributed graphs.

The CODA algorithm [12] detects contexts and, consequently, the nodes that have not been assigned to any context as outliers. CODA results are binary: a node is either an outlier or an inlier, that is, CODA does not rank the outliers. To perform a ranking and also to identify outliers in subspaces, ConSub [36] statistically selects congruent subspaces. GOutRank [27] is based on clustering algorithms in subspaces and scores nodes according to their membership in multiple subspace clusters. FocusCO [30] detects clusters and corresponding outliers in a user-driven manner.

ConOut [35] assigns each node to a single context (subgraph) and its statistically relevant subset of attributes. As we use this method in the experiments in this work, we describe its central ideas in more detail. The context selection step aims to find local neighborhoods that are similar with respect to the graph structure. For example, if two nodes share a large number of neighbors, they should belong to the same context. Overall, the context of a node is the reflexive transitive closure of adjacent nodes that are significantly similar to the first. The next step is to select attributes for each context through the comparison of the distribution of all attribute values in the local context to the whole dataset distribution. A statistical test (F-test or Kolmogorov-Smirnov test) checks whether the context values present a significantly smaller variance compared to the entire dataset

distribution. The attributes are locally chosen according to this test. The anomaly score is finally based on the multiplication of a local graph density and a local attribute deviation. These two values represent structural deviations (e.g., a node that does not present the same structural pattern as its context members) and attribute deviations (e.g., a node that has different attribute values than its context members).

A recently proposed approach named Radar [22] detects outliers by modeling attribute and network information from a residual analysis perspective. The idea behind Radar is that the attributes of every instance can be reconstructed by a linear combination of some representative instances, when considering only the information perspective. Additionaly, when link information between instances is added to the model, the framework relies in a homophily property, such that similar instances are more likely to be linked together. The authors propose an optimization problem where the objective function should be minimized to find the best coefficient matrix $\mathbf{W}$ and residuals $\mathbf{R}$ for the reconstruction of attributes whithin the network context. At the end of the optimization process, the anomaly score for each instance i is computed as the ℓ_2-norm of the i-th row of the residual matrix $\mathbf{R}$.

All techniques of anomaly detection in static attributed graphs are performed on a single graph representation derived from a given dataset. Normally, there are no works that assess the impact of the graph modeling process to a single graph for real-world databases. There are also no works that discuss the pros and cons of employing multiple graphs models to represent a given database.

2.2. Ensemble Techniques in Outlier Detection

Outlier detection in general has been improved by using ensemble methods, i.e., combining the findings or results of individual learners to an integrated, typically more reliable and better result. An ensemble is expected to improve over its components if these components deliver results with a certain minimum accuracy while being diverse [42]. The two main challenges for creating good ensembles are, therefore, (i) the generation of diverse (potential) ensemble members and (ii) the combination (or selection) of members to an ensemble.

Some strategies to achieve diversity among ensemble members are feature bagging (i.e., combining outlier scores learned on different subsets of attributes) [20], different parameter choices for some base method [11], the combination of actually different base methods [28,19,37], the introduction of a random component in a given learner [24], the use of different subsamples of the data objects [44], adding some random noise component to the data ("perturbation") [43], or using approximate neighborhoods for density estimates [15]. In a sequential setting, the first top outliers detected are removed from the data set before it is handed over to other learners [33].

Different combination procedures have been proposed based on outlier scores or on outlier rankings [20,11,19,42,32]. Some methods have also been proposed to select the more diverse or (in a semi-supervised setting) the more accurate ensemble members [37,25,33,32].

All ensemble methods in outlier detection aim at the reduction of bias and variance inherent to some learner. Assessing the impact of the bias inherent to the input data has not been addressed so far.

Considering the standard KDD process model [10], our focus is on the data transformation and its impact on what we can learn from the data. The ensemble approach is thus effectively moved to an earlier step in the KDD process, since we employ different graph models to represent the same raw data.

3. Methodology

The classical approach for detecting outliers in a dataset is to model the data as a single graph and to apply a single outlier detection method, as sketched in Figure 1(a). By using this approach, the identification of outliers is biased by the given model and the selected algorithm. Alternatively, one could use an ensemble approach to apply a set of complementary outlier detection methods on a single graph and combine their results, such that the algorithm bias is reduced. This approach is sketched in Figure 1(b). Existing work for outlier detection in graphs follows the methodologies in Figures 1(a) and 1(b). As a consequence the built-in bias from the graph model selection is not adressed.

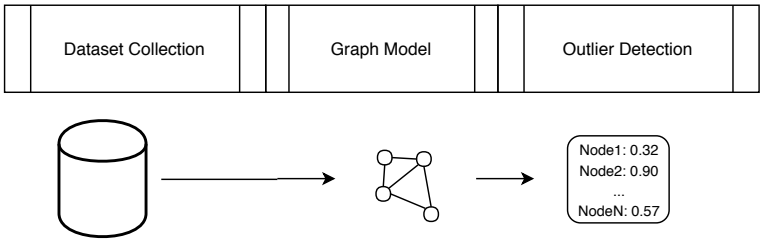
Here we propose a new methodology that tackles the reduction of graph model bias towards outlier detection by generating multiple graph models to represent the same data. The overall workflow for an ensemble method combining outlier detection results from multiple graphs is depicted in Figure 1(c). First, multiple graph models represent the same dataset, possibly taking different aspects of the dataset into account for deriving different graph models. We assume, though, that the nodes in different graphs represent the same entities. Only their relations change from model to model. Next, some algorithm to detect (node) outliers in graphs are applied to each graph model. In the last step, results from the outlier detection on the different graph representations are combined. Through the ensemble of different graphs modeling the same data, we can expect an increasing precision and robustness of the outlier detection.

For this general approach, various questions could be studied, for example, which outlier detection methods are more suitable and how the results should be combined. For this study, however, we take a fixed decision on these questions, using two methods for outlier detection and using four consensus functions to combine the results, as we are studying the impact of the design and choice of graph models.

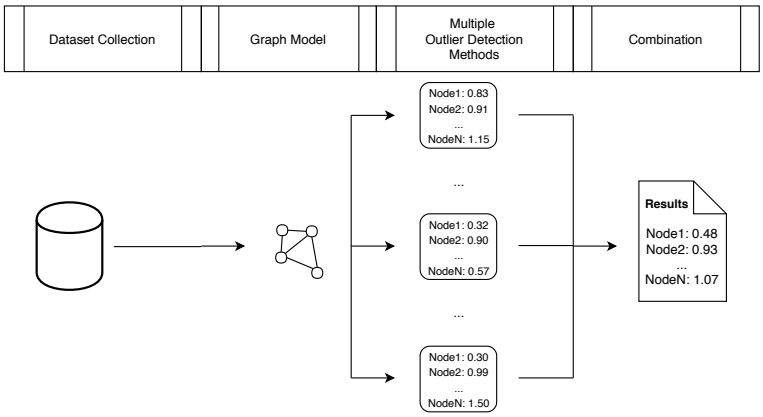
We describe two methodologies employed in the experimental assessment presented in this work. The first methodology aims at assessing the potential gains of combining *multiple* graph models compared to using a *single* graph model. The second methodology aims at assessing the improvements for outlier detection through the combination of *complementary* multiple graph models that capture preferably different aspects of the data.

3.1. Characterizing Multiple Graph Models

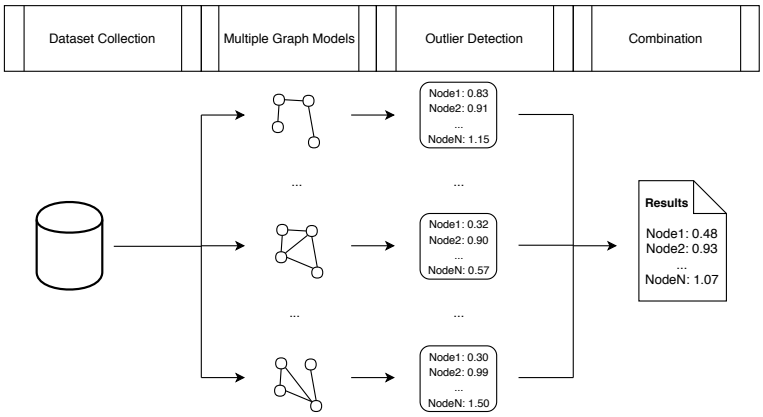
We use multiple graph models here to represent different aspects of a data set, i.e., to take different perspectives. For all perspectives, i.e., derived graph models, for some dataset, the entities that may be outliers remain the same and are vertices in the graph. The different perspectives are expressed by different edges describing relations between the nodes. How different graph models express different aspects of a dataset depends on the dataset and its semantic. We will discuss specific graph models for different example datasets later. Here we characterize the notion of multiple graph models more formally.



(a) Normal outlier detection workflow on graphs.



(b) Ensemble of multiple outlier detection methods on a single graph model.



(c) Multiple graph models for outlier detection.

Fig. 1. Different workflows for outlier detection in graphs.

Definition 1. ***(Graph Model)***
A graph model G_p is described by $G_p = \{V, E_p, A\}$, where

- *p is the perspective of the graph G_p.*
- *V represents the node set and each node $v \in V$ represents an entity.*
- *E_p represents the edge set, where $E_p = \{(v_i, v_j) \mid \forall i \neq j, v_i, v_j, \in V \wedge \mathcal{F}_p(v_i, v_j) \geq t_p\}$. The function $\mathcal{F}(.)$ returns a score $\in [0, 1]$ that defines for a given perspective p the correlation or similarity between two nodes, and t is a user-defined threshold.*
- *A is the attribute set and each attribute $a \in A$ represents an attribute.*

Multiple graph models as outlined above can then be characterized as follows:

Definition 2. ***(Multiple Graph Models)***
A set of multiple graph models $\mathcal{G} = \{G_1, G_2, ..., G_p, ..., G_M\}$, where M is the total amount of perspectives, we have:

$$V_i = V_j \forall i, j \in [1, 2, ..., M].$$

Our definition assumes that different graph models have the same node set V. We thus avoid the node-to-node mapping problem, since the set of vertices, related to the dataset entities that may turn out being outliers, remains the same across the different graph models.

Given a dataset D, we generate multiple graph models $\mathcal{G} = \{G_1, G_2, ..., G_p, ..., G_M\}$ with respect to different perspectives p of M. Suppose we have two graphs $G_1 = \{V, E_1, A\}$ and $G_2 = \{V, E_2, A\}$. A single perspective p relates to a specific procedure to generate the edges of G_p, though possibly with different parametrization. For example, if D represents a citation network dataset, p is co-authorship, G_p will represent D as a co-authorship graph, in which nodes represent authors and edges represent the existence of co-authorship. In this example, an edge $e_{ij} = (v_i, v_j)$ will exist iff $\mathcal{F}(v_i, v_j) = 1$, where $\mathcal{F}(.)$ returns 1 if v_i and v_j have at least one publication together and 0 otherwise.

The perspective p may also represent correlations or similarities between nodes. For example, in a citation network dataset, a graph G_p may be defined as the correlation between publications in conferences based on co-occurrence of words in the title or in the abstract, where two nodes (authors) are connected if they have large correlation values. Or a perspective may be the defined as the similarity between authors in their research topics. In these scenarios, an edge $e_{ij} = (v_i, v_j)$ will exist iff $\mathcal{F}(v_i, v_j) > t_p$, where t_p denotes the correlation or similarity threshold in perspective p.

As these examples demonstrated, the definition of a perspective depends on the given data for a given problem. We will introduce tailored perspectives on the various experimental datasets in the case studies (Section 4).

3.2. Characterizing the Gains of Using Multiple Graph Models

To characterize the gains of using multiple graph models, we propose the following steps:

1. **Generate multiple graph models according to the problem being addressed.** For each raw dataset, we design multiple graph models that describe the same entities as nodes but differ quantitatively (how dense they are) and qualitatively (which relationships are expressed in the graph structure). These multiple graph models aim to

materialize the various perspectives that one can take on the data. As a result, they can make different kind of outliers detectable.
The design of graph models for some raw data set remains problem-dependent, though. We explore different variants for synthetic datasets and for real datasets.

2. **Assess experimentally the combined usage of multiple graph models.** We quantify experimentally the gains of multiple graph models using both synthetic and real-world datasets.

3.3. Selective Composition of Multiple Graph Models

Orthogonal to the generation of as many graph models as practical is the design of graph models that are as complementary as possible, i.e., they should model different aspects of the raw data. The less similar two graphs are, the more likely they are complementary regarding the outlier information they provide.

Two different graphs $G_1 = (V, E_1, A)$ and $G_2 = (V, E_2, A)$, which differ only in their set of edges E_1 and E_2, can have a degree of similarity measured by Jaccard [14] on their sets of edges:

$$Jaccard(G_1, G_2) = \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|}. \tag{1}$$

The similarity of a set $\mathcal{G}$ of more than two graphs is assessed as the average pairwise similarity:

$$Jaccard(\mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\substack{G_1, G_2 \in \mathcal{G} \\ G_1 \neq G_2}} Jaccard(G_1, G_2). \tag{2}$$

DeltaCon [17] is another algorithm to compute similarity between graphs. Given two input graphs $G_1 = (V, E_1, A)$ and $G_2 = (V, E_2, A)$, with different edge sets E_1 and E_2, DeltaCon computes the similarity score $DeltaCon(G_1, G_2) \in [0, 1]$, such that a score of 1 means identical graphs.

DeltaCon applies the Fast Belief Propagation (FaBP) [16] method to measure node affinity in the same graph and to build the $n \times n$ similarity matrix $\mathbf{S}$, based on the $n \times n$ identity matrix $\mathbf{I}$, the $n \times n$ diagonal degree matrix $\mathbf{D}$, and the $n \times n$ adjacency matrix $\mathbf{A}$:

$$\mathbf{S} = [s_{ij}] = [\mathbf{I} + \epsilon^2 \mathbf{D} - \epsilon \mathbf{A}]^{-1} \tag{3}$$

ϵ is a small constant to regulate influence of the neighboring nodes.

The main idea is to analyze graph differences from the information flow viewpoint. For instance, a missing edge in a clique subgraph is not as important as a missing edge connecting two dense subgraphs.

In its fastest version, DeltaCon divides the node set into g groups and computes the affinity score of each node to each group. The $n \times g$ similarity matrix S_1 and S_2 is then built for each graph G_1 and G_2, respectively, and the similarity score is computed as follows:

$$d(G_1, G_2) = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{g} \left(\sqrt{s_{1,ij}} - \sqrt{s_{2,ij}}\right)^2} \tag{4}$$

$$DeltaCon(G_1, G_2) = \frac{1}{1 + d(G_1, G_2)} \tag{5}$$

The similarity of a set $\mathcal{G}$ of more than two graphs is assessed as the average pairwise similarity:

$$DeltaCon(\mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{\substack{G_1, G_2 \in \mathcal{G} \\ G_1 \neq G_2}} DeltaCon(G_1, G_2). \tag{6}$$

While previous methods to select members for outlier detection ensembles (trained on the same dataset) rely on some sort of estimate of the quality in the unsupervised scenario [37,32], the similarity of graph models is completely agnostic of (supposed) outliers and can therefore serve as a general guidance to the selection of ensemble members.

4. Implementation of the Methodology: Case Studies on Synthetic and Real Data

We perform case studies on synthetic data, data from DBLP, data from Citation Network, and data from Facebook. We focus on quantitatively different graphs for synthetic data, and on semantically different graphs for DBLP, Citation Network, and Facebook data. These real-world databases were chosen to represent realistic scenarios where multiple graph models are viable ways to help detecting outliers. By considering different areas we demonstrate the reproducibility of our results. It should be noted that it is straightforward to derive quantitatively different graphs in an automated way, while semantically different graphs need to be designed manually for each problem, taking into account the semantics of the data and considering which information may be possibly interesting.

4.1. Synthetic Datasets

For a quantitative experiment, we generate three families of synthetic datasets (A, B, and C) to evaluate the benefits of combining multiple graph models from the same source: (A) graphs following a power-law degree distribution, in particular Zipf's law according to Gao et al. [12], (B) graphs generated by following a stochastic algorithm proposed by Barabási and Albert [3], and (C) graphs following the Erdős and Rényi model [9], in which every possible edge is created with the same constant probability. For these dataset families, we start with a graph comprising 4950 nodes generated according to the corresponding distribution or algorithm, respectively, and include 50 outliers that follow a uniform degree distribution. We repeat this procedure 10 times, so we have 10 base graphs for each experiment with 5000 nodes each, where the inliers follow a known degree distribution and outliers follow a uniform degree distribution. We also add 20 attributes and set a random percentage of those to be relevant for the outlier detection task. Relevant attributes follow different Gaussian distributions for outliers and inliers with $\mu = [-10, 10]$ and $\sigma = [1, 5]$. Irrelevant attributes follows the same Gaussian distribution for outliers and inliers.

We use two parameters to induce diversity in the graphs, determining the percentage of edges that we remove from the graph or that we add to the graph, respectively. This way we achieve diverse graph structures without changing the number of nodes or their attribute values, by randomly removing and adding edges. We derive 10 graphs of different density from each of the 10 base graphs in graph families A, B, and C, resulting in 100 graphs for each experiment.

Table 1. Number of authors per publications at 23 conferences.

Number of Publications	Number of Authors
1	44905
2	11940
3	5398
4	3144
5	2078
6-10	4421
11-20	2282
> 20	1488
78350	**75656**

4.2. DBLP Data

The bibliography dataset provided by DBLP (`http://dblp.uni-trier.de/db/`) is a rich dataset comprising several informations about publications in the computer science area. For our assessment, we sample a dataset containing just the publications from 23 related conferences, namely: AAAI, IAAI, CIKM, CVPR, ECIR, ECML, PKDD, EDBT, ICDT, ICDE, ICDM, ICML, IJCAI, KDD, PAKDD, PODS, SDM, SIGIR, SIGMOD, SSDBM, VLDB, WSDM, and WWW. We select authors with at least 5 publications to compose the node set of our graph models, leading to 10269 nodes. In addition, we describe each author through 47 different attributes that represent the ratio of the author's publication in each conference over the total amount of the author's publication (23 attributes), the ratio of the author's publication in each conference over the total amount of conference publications (23 attributes), and the normalized amount of publications (1 attribute).

Table 1 shows a relation between number of publications and number of authors. Since we selected only authors that have at least 5 publications, this amounts to 13.57% of the DBLP dataset regarding the 23 conferences mentioned. Figure 2 shows for each conference the number of publications and unique authors. CVPR conference has the highest amount of publications, but AAAI has the highest amount of unique authors. On the other side, IAAI has the lowest amount of publications and ICDT the lowest amount of unique authors.

As we select authors that have at least 5 publications in the 23 conferences, our modeled graphs may present scenarios in which a vertex has no connections, i.e., it is an isolated node in the graph. Consider, for example, that one author has 5 publications and no co-authorship. In this case, the author would be represented by an isolated node in the co-authorship graph. We consider such isolated nodes as ranked last since the outlier detection algorithms used do not handle such scenarios with isolated nodes.

While we focused on quantitatively different graphs in the synthetic data, here we derive four *semantically different* graph models from the DBLP data:

1. Co-authorship: If two authors have at least one publication together, they will have an edge connecting them.
2. Correlation between publications in each conference: Two authors are connected if they have a high correlation of their distribution of publications over the conferences (i.e., they tend to publish at the same venues).

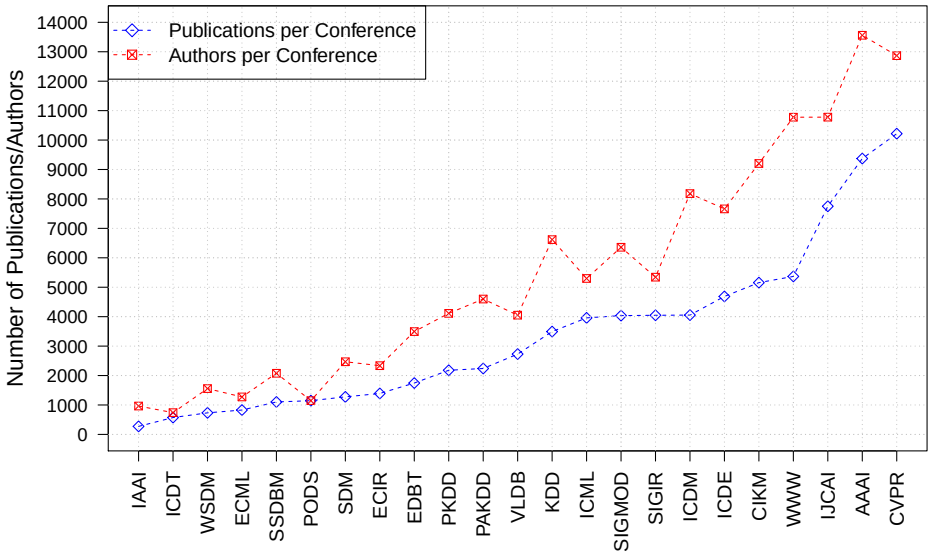


Fig. 2. Authors and publications per conference.

3. Correlation between publications in each area of knowledge: In order to model knowledge areas, we group conferences into five areas: Artificial Intelligence, Data Mining, Databases, Information Retrieval, and Web. In this model, two authors are connected if they have a high correlation between publications in venues from similar areas of knowledge.

   A correlation threshold may be seen as a parameter of graph density: the lower the threshold, the denser the graph. In models 2 and 3, we used 0.9 as correlation threshold to determine the existence of an edge.

4. Similarity in topics: Two authors are connected if they have publications with similar title words. We determine the set of unique words used in all publications of each author, removing all stop words, applying Porter's stemming algorithm [31], and selecting all unique words for each author. The edge between two authors exists if the overlap between their vocabularies is at least 40%, that is, if the intersection between their title words is at least 40% of the smaller of both vocabularies.

For sake of quantitative evaluation, we add 50 artificial outliers by randomly selecting 50 times 2 to 5 nodes that we merge to generate one outlier. Such merging imitates the effect of having two (or more) authors with the same name in the database without name-disambiguation such that their publication profiles are merged in DBLP, which is a quite realistic problem. These merged nodes are the same across all graph models. We repeat the artificial outlier generation 10 times, resulting in 10 DBLP datasets with different outliers.

4.3. Citation Network Data

The Citation Network data was extracted from ACM and provided at `https://aminer.org/citation` [39]. It comprises information about publications in many venues. For each paper, it provides id, title, abstract, year of publication, authors, venue, and references. The main difference between DBLP and Citation Network is that in the first one the node set represents authors, while here nodes are papers.

We preprocessed the original Citation Network ACM version 09 dataset, which was collected until 2017 by removing publications with missing information, selecting authors with at least 10 publications, and conferences with at least 500 papers to compose the node set of our graph models. After this pre-processing step, the remaining 16,197 papers represent our Citation Network dataset.

In addition, we describe each paper through 6 different attributes derived from the information given on the original dataset: year of publication, average authors per publications, average authors per publications in the conference where the paper was published, number of conference papers accepted in that year, and total amount of papers in the conference.

We derive four *semantically different* graph models from the Citation Network data:

1. Reference: Connect two papers if at least one of them references the other.
2. Co-authorship: If two papers share at least one author, they are connected by an edge.
3. Similarity in abstract: Similar to the DBLP model 4, we determined the set of unique words used in each paper's abstract, removed all stop words, applied Porter's stemming algorithm [31], and selected all unique words for each paper. If two papers share at least 40% of their abstract vocabulary, we connect them with an edge.
4. Similarity in topics: This is the same as DBLP model 4. We perform the same approach for word processing and connect two papers if they share at least 40% of their title words.

Here we include 50 realistic outliers to simulate effects of plagiarism. We selected 50 different papers and copied their title and abstract information, changing the authors, the venue and the year. To select the authors and the venue, we choose by random from the pool of possible choices that is present in our preprocessed dataset. The year we select randomly from the orginal publication until 2017. We repeat the artificial outlier generation 10 times, resulting in 10 Citation Network datasets with different outliers.

Each paper has attributes that are derived from the year, the authors, and the venue. The values of these attributes are suspicious for outliers as we generate them by selecting authors, venue, and year randomly. We expect that different graph models change the vicinity of outlier nodes in different aspects, so that the 'plagiarized' papers are considered a more significant outlier in some models. For example, in a graph where papers are connected by similar title words, the authors that plagiarized a paper should have a different behavior of publications than other authors that published in similar topics.

4.4. Facebook Data

We use the Facebook dataset provided by Leskovec and Krevl [21]. This dataset consists of 10 ego nodes and their friends from Facebook. Each ego and its friend list have a set of attributes related to, for example, age, gender, education, or work. We selected the 3

largest ego nodes (ids 107, 1684, and 1912) and their friends to generate the dataset used in this experiment. This was done in order to group nodes that have similar sets of attributes, since not all 10 ego nodes have the same attribute set. This dataset has 2,573 nodes in total. The IDs and feature vectors of this dataset have been anonymized. 88 attributes describe aspects such as education, work, gender, location, language, and birthday.

We derived 3 semantically different graphs to represent this dataset:

1. connecting people according to their friendship,
2. connecting people that have correlated education features, and
3. connecting people that have correlated work-related features.

As for the DBLP models 2 and 3, we use 0.9 as correlation threshold.

The outlier generation process was performed the same way as for the DBLP dataset, randomly selecting 2 to 5 nodes to merge and to generate an outlier. We added 50 outliers to the dataset, repeating the random procedure 10 times, resulting in 10 Facebook datasets with different outliers.

4.5. Outlier Detection Methods

To perform outlier detection on each individual graph, we use ConOut [35] and Radar [22]. On ConOut, we use similarity parameter 0.5, and a significance level parameter 0.1 as suggested by Sánchez et al. [35]. The similarity parameter indicates the threshold to consider two nodes similar or not. The significance parameter indicates whether we accept the statistical F-test or reject it (as described on Section 2.1). In other words, if the p-value is below the significance parameter, the attribute has lower variance inside the context comparing to the whole dataset, thus we add it to the set of relevant attributes.

On Radar, we use $\alpha = 0.5$, $\beta = 0.2$ and $\gamma = 0.2$ as suggested by the authors [22]. These parameters control the row sparsity of the coefficient matrix, row sparsity of the residual matrix and the contribution of the network modeling, respectively.

ConOut and Radar are examples of outlier detection methods in static graphs that output scores of outlierness for each node. This type of methods is suitable to our scenario as we, in a posterior step, combine these outputs into a single score. Any other outlier detection method may be used after adjusting the combination technique.

4.6. Combination

To combine the results obtained from different graph models derived from the same dataset into a single outlier score for each node we use two rank and two score combination procedures. In this particular scenario, we do not need a normalization step for score combination since we apply the same algorithm with same parameters on different graphs. We combine scores by taking the average value and the maximum value.

We also transform the scores into rankings and aggregate them by using the median and the Borda Count algorithm [4]. The Borda Count method is a classic voting system that can be applied to combine rankings from different sources. This method gives a score to each member of the list according to its relative position. The final aggregated ranking is a simple sorted list based on the sum of scores of each member. This is equivalent to combining the rankings by their mean rank.

Table 2. Average ROC AUC values on Power Law (Exp. A) synthetic data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	0.633				**0.655**			
2	0.671	0.634	0.668	0.668	0.652	0.651	0.654	0.654
3	0.707	**0.638**	0.681	0.707	0.651	0.651	**0.657**	0.655
4	0.728	0.634	0.719	0.732	0.651	0.651	**0.657**	**0.656**
5	0.746	0.627	0.729	0.754	0.651	0.651	**0.657**	0.655
6	0.749	0.612	0.744	0.759	0.651	0.651	**0.657**	0.655
7	0.756	0.612	0.742	0.766	0.652	0.651	0.656	0.655
8	**0.757**	0.600	**0.753**	**0.767**	0.652	0.651	0.656	0.655
9	**0.757**	0.590	0.750	**0.767**	0.652	0.651	0.656	0.655
10	0.748	0.580	0.742	0.761	0.652	0.651	0.655	0.654
Ens. Size	DeltaCon Similarity							
1	**0.633**				**0.655**			
2	0.672	0.630	0.681	0.681	0.652	0.651	0.653	0.653
3	0.688	0.580	0.668	0.699	0.650	0.651	0.651	0.652
4	0.705	0.577	0.701	0.717	0.650	0.651	0.654	0.654
5	0.714	0.577	0.703	0.724	0.651	0.651	**0.655**	0.654
6	0.721	0.580	0.716	0.733	0.651	0.651	**0.655**	0.654
7	0.728	0.576	0.718	0.740	0.651	0.651	**0.655**	0.654
8	0.736	0.581	0.729	0.748	0.651	0.651	**0.655**	0.654
9	0.741	0.580	0.725	0.752	0.651	0.651	**0.655**	0.654
10	**0.748**	0.580	**0.742**	**0.761**	0.652	0.651	**0.655**	0.654

To assess the impact of graph similarity on the quality of the combined results we use Jaccard similarity (Equation 1) and DeltaCon similarity (Equation 5). Combinations of more than two graph models are ranked by the average pairwise Jaccard similarity (Equation 2) and average pairwised DeltaCon similarity (Equation 6).

5. Results and Discussion

5.1. Synthetic Data

The results for the synthetic datasets are shown in Tables 2, 3, and 4 for experiments A, B and C respectively. These tables shows quantitatively the impact on outlier detection performance if we take the most dissimilar models to combine the results obtained on those. The ensemble size is the number of combined graph models, where 1 relates to a single graph model (i.e., no combination but individual performances on all graphs) and 2 to 10 relate to the corresponding number of combined multiple graph models. The performance is quantified in terms of the average Area Under the Curve of the Receiver Operating Characteristic (ROC AUC), a standard measure for outlier detection performance [5].

In all three experimental scenarios, using multiple graph models generally improves over using individual graphs only. Radar seems to be a more robust method than ConOut

Table 3. Average ROC AUC values on Barabási and Albert (Exp. B) synthetic data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	0.647				**0.762**			
2	0.792	0.757	0.751	0.751	0.753	0.753	0.758	0.758
3	0.813	0.749	0.762	0.778	0.754	0.753	0.762	0.760
4	0.835	**0.760**	0.793	0.806	0.754	0.753	0.762	0.760
5	0.842	0.757	0.786	0.815	0.754	0.753	**0.763**	0.761
6	0.849	0.755	0.807	0.823	0.755	0.753	**0.763**	0.761
7	0.855	0.754	0.802	0.830	0.755	0.753	**0.763**	0.761
8	0.858	0.754	0.813	0.832	0.755	0.753	**0.763**	**0.762**
9	0.861	0.755	0.811	0.835	0.755	0.753	**0.763**	**0.762**
10	**0.863**	0.756	**0.814**	**0.837**	0.755	0.753	**0.763**	**0.762**
Ens. Size	DeltaCon Similarity							
1	0.647				**0.762**			
2	0.784	0.747	0.752	0.752	0.753	0.753	0.758	0.758
3	0.825	0.762	0.769	0.797	0.753	0.753	0.761	0.759
4	0.830	0.758	0.784	0.797	0.754	0.753	0.762	0.760
5	0.845	**0.760**	0.786	0.817	0.754	0.753	0.762	0.761
6	0.850	0.758	0.806	0.825	0.754	0.753	0.762	0.761
7	0.852	0.759	0.793	0.825	0.755	0.753	**0.763**	0.761
8	0.855	0.757	0.806	0.827	0.755	0.753	**0.763**	**0.762**
9	0.858	0.757	0.801	0.831	0.755	0.753	**0.763**	**0.762**
10	**0.863**	0.756	**0.814**	**0.837**	0.755	0.753	**0.763**	**0.762**

as its results have lower variation. It is out of our scope to analyze different parameter values for the outlier detection algorithms, as we are interested on the effects of applying multiple graph models on the same raw dataset. As mentioned before, the γ parameter on Radar balances the contribution of attribute and network information. On our experiments, we fix γ as suggested by the authors and, as a consequence, the results show robustness towards multiple graph models, as the variations on the edges do not have a high impact on the final Radar output.

ConOut is largely benefited by the usage of multiple graph models. Using ConOut on single graphs, average ROC AUC reaches 0.633, 0.647, and 0.637 on experiments A, B and C respectively. In comparison with multiple graph models, these numbers go as high as 0.767, 0.863, and 0.814. On most cases, combining all 10 outputs from different graphs achieves the highest ROC AUC value, which indicates that using multiple graph models has large potential to be applied in many different scenarios, as we have 3 different synthetic experiments that express real-world graph behaviors.

Regardless of the algorithm applied, combine scores using the maximum value do not perform well in general. Experiment C using Radar algorithm is the only specific scenario where combining by maximum has the best average ROC AUC value, 0.712. Mean, median and borda have a very similar behavior when used in conjunction with the

Table 4. Average ROC AUC values on Erdős and Rényi (Exp. C) synthetic data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

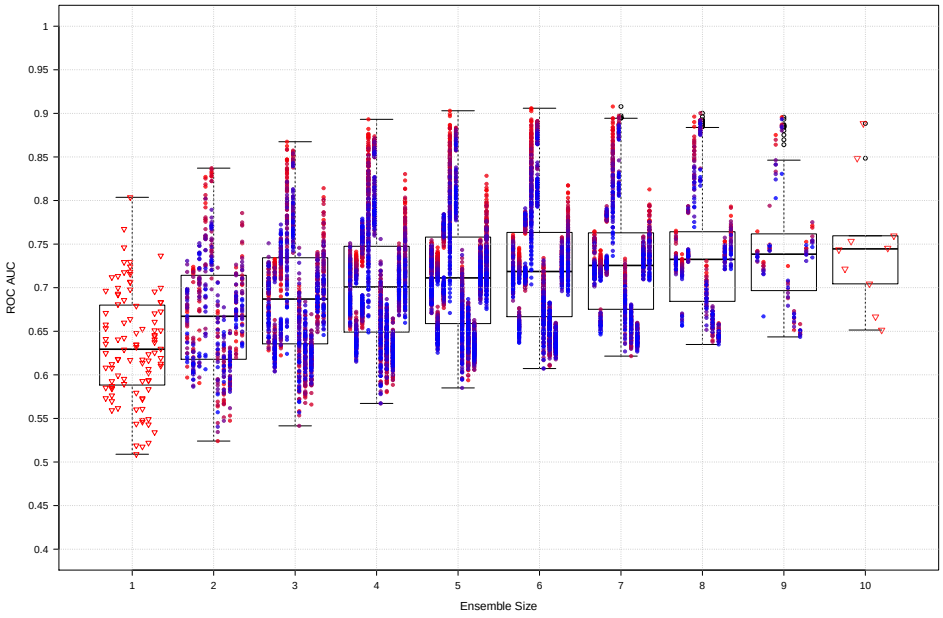
Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	0.637				**0.701**			
2	0.753	0.720	0.727	0.727	0.705	0.707	0.700	0.700
3	0.750	0.696	0.713	0.726	0.703	0.704	0.695	0.698
4	0.783	0.717	0.749	0.758	0.703	0.705	0.697	0.699
5	0.798	0.726	0.740	0.770	0.703	0.705	0.697	0.698
6	0.805	0.726	0.760	0.778	0.703	0.705	0.697	0.698
7	0.807	0.734	0.759	0.780	0.704	0.707	0.697	0.699
8	0.808	0.739	**0.766**	**0.781**	0.704	0.707	0.698	0.699
9	**0.814**	0.750	0.760	**0.781**	0.706	0.711	0.698	0.699
10	0.807	**0.758**	0.758	0.774	**0.707**	**0.712**	0.699	**0.701**
Ens. Size	DeltaCon Similarity							
1	0.637				**0.701**			
2	0.741	0.712	0.718	0.718	0.705	0.706	**0.701**	0.701
3	0.778	0.731	0.727	0.751	0.707	0.709	**0.701**	**0.702**
4	0.791	0.732	0.754	0.761	**0.708**	0.710	0.699	0.701
5	0.807	0.748	0.754	0.778	**0.708**	0.711	0.700	**0.702**
6	0.806	0.740	0.762	0.776	0.707	0.711	0.699	0.701
7	0.812	0.748	0.758	**0.782**	0.707	0.711	0.698	0.700
8	0.809	0.752	**0.765**	0.779	**0.708**	**0.712**	0.698	0.701
9	**0.813**	0.756	0.758	**0.782**	**0.708**	**0.712**	0.698	0.701
10	0.807	**0.758**	0.758	0.774	0.707	**0.712**	0.699	0.701

ConOut algorithm. These combination approaches yield their highest values using similar ensemble sizes on each experiment, which again supports the argument for using multiple graph models rather than single graphs.

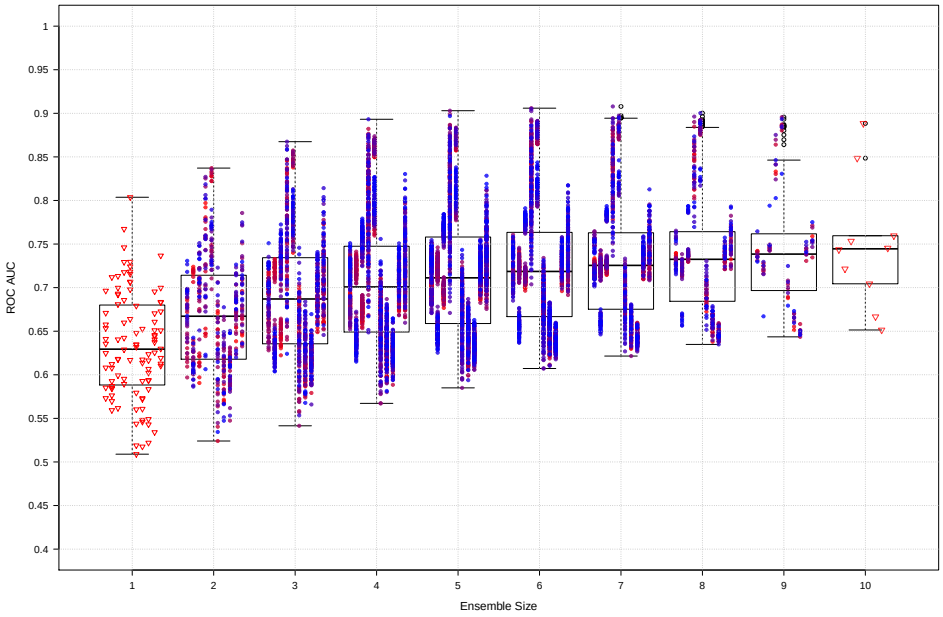
The results do not change much between using Jaccard similarity and DeltaCon similarity.

From Tables 2, 3, and 4 we can infer that using the most dissimilar models according to Jaccard and DeltaCon to represent the same dataset in general improves the performance on detecting outliers. Figures 3, 4 and 5 shows results using ConOut on synthetic experiments. The colors of the points represent how similar the combined graph models are, ranging from red to blue, where blue is more similar and red is less similar. Each experiment is represented by one plot using Jaccard similarity and one plot using DeltaCon similarity. We expect more red points on the top of each boxplot, which means that the combination of dissimilar graphs delivers better results than combining similar graphs. Since we have 10 independent subsets for each graph family, each column of points inside the boxplots shows the results for one of these subsets.

All three experiments show more red dots on the top of boxplots, especially when Jaccard is selected. The behavior of boxplots reinforces our claim on using multiple graph

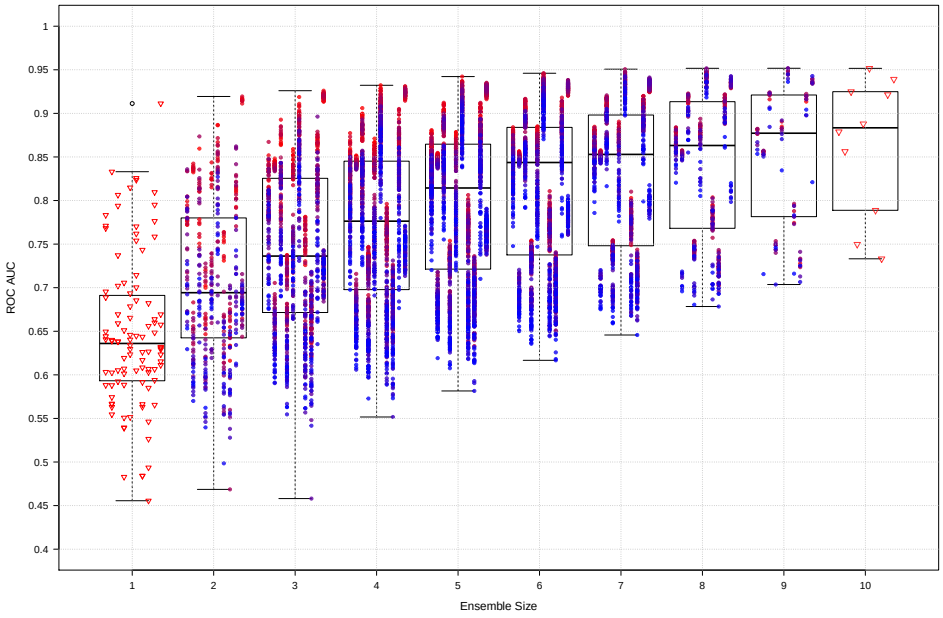


(a) Experiment A: Power-law degree distribution with ConOut using Jaccard similarity measure

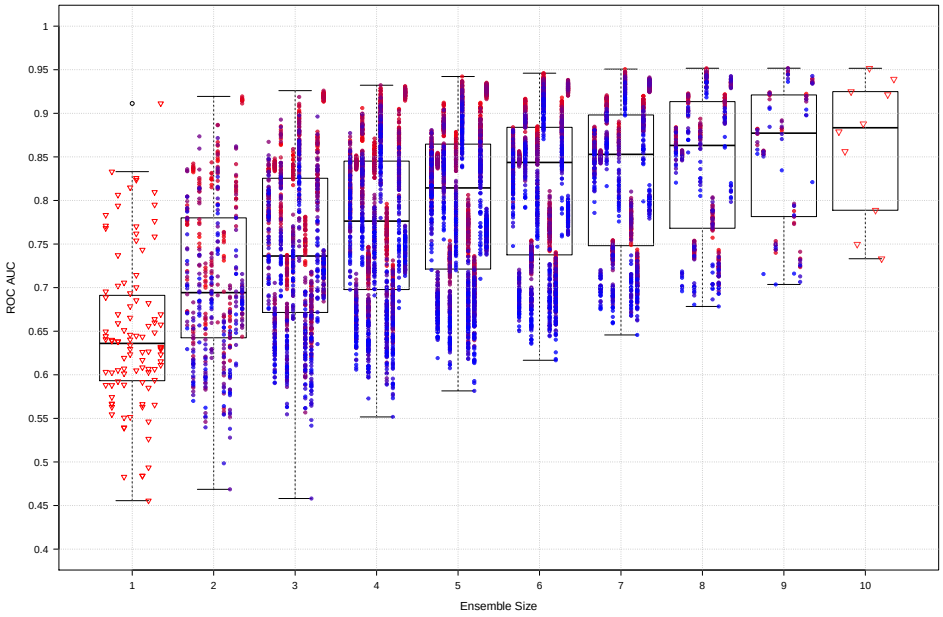


(b) Experiment A: Power-law degree distribution with ConOut using DeltaCon similarity measure

Fig. 3. ROC AUC on single and multiple graph models on experiment A synthetic datasets as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.

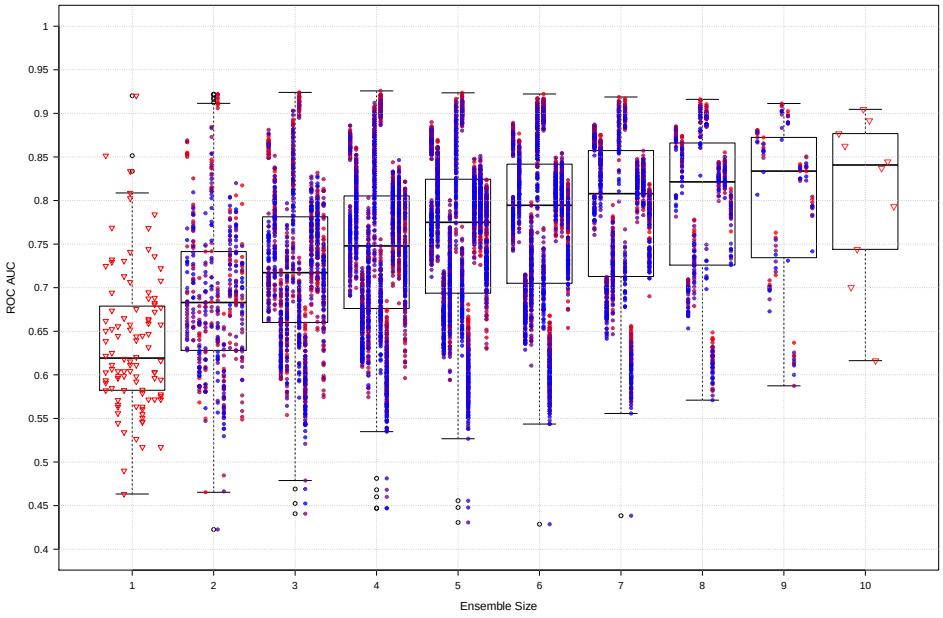


(a) Experiment B: Barabási and Albert stochastic model with ConOut using Jaccard similarity measure

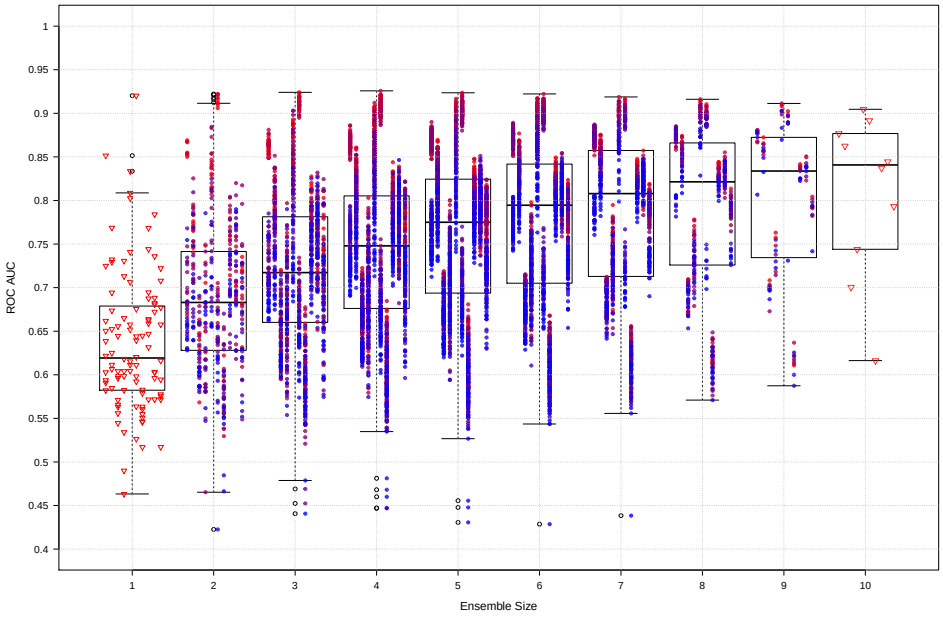


(b) Experiment B: Barabási and Albert stochastic model with ConOut using DeltaCon similarity measure

Fig. 4. ROC AUC on single and multiple graph models on experiment B synthetic datasets as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.



(a) Experiment C: Erdős and Rényi model with ConOut using Jaccard similarity measure



(b) Experiment C: Erdős and Rényi model with ConOut using DeltaCon similarity measure

Fig. 5. ROC AUC on single and multiple graph models on experiment C synthetic datasets as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.

Table 5. Average ROC AUC values on DBLP data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	0.840				**0.506**			
2	0.890	0.874	0.896	0.896	0.501	**0.510**	0.511	0.511
3	0.898	0.877	0.888	0.903	0.505	0.505	0.497	0.504
4	**0.953**	**0.934**	**0.939**	**0.943**	0.495	0.499	**0.527**	**0.527**
	DeltaCon Similarity							
1	0.840				**0.506**			
2	0.890	0.874	0.896	0.896	0.501	**0.510**	0.511	0.511
3	0.950	**0.934**	0.927	0.933	0.502	0.494	**0.528**	**0.534**
4	**0.953**	**0.934**	**0.939**	**0.943**	0.495	0.499	0.527	0.527

models for outlier detection, as the ROC AUC increases when we select a larger ensemble size.

Overall, using multiple graph models achieves better performance than using single graph models, regardless of the similarity parameter threshold, of the number of models, and of which models are combined: the tendency of the results shows that the performance on multiple graph models is correlated to the performance of the individual models we combine.

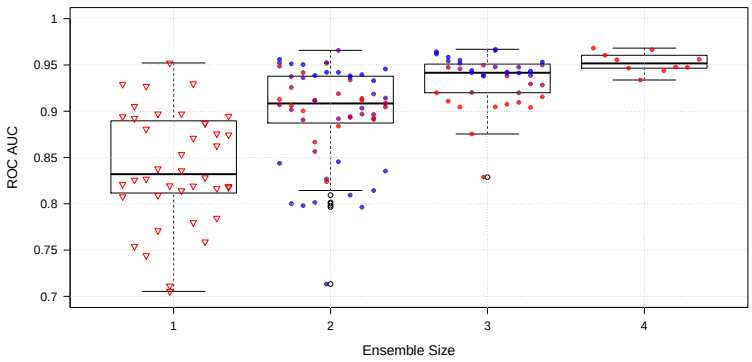
Multiple graph models represent different aspects of the data, where each model potentially highlights different outliers. When we combine the outlier detection results obtained over multiple graph models we benefit from this diversity.

5.2. DBLP Data

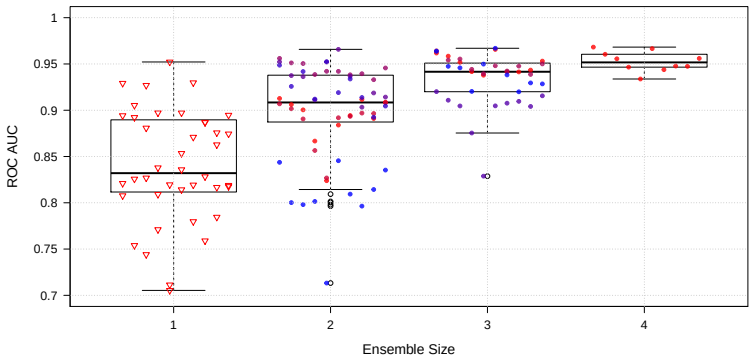
Table 5 shows the average ROC AUC results for ConOut and Radar on single and multiple graph models. We again see the robustness of the Radar algorithm. Even though Radar performs poorly on DBLP data, using multiple graph models for outlier detection improves over single graphs. ConOut reaches average ROC AUC of 0.953 in the best scenario combining all graph models outputs using mean, in comparison to 0.840 when using only single graphs to represent the data.

Figure 6 shows the results when we apply ConOut to single graph models (ensemble size 1) and to multiple graph models, using the mean as consensus function.

Although the quantitative results for the DBLP dataset are quite good in terms of precision, there are still some authors that consistently get high outlier scores together with the generated outliers (merged nodes). For the quantitative evaluation based on the artificially constructed outliers, these are considered inliers, but they might actually qualify as real outliers in some sense. Thus we inspect some examples in Figure 7, depicting the degree of outlierness for 9 authors that consistently present high scores. These plots measure the degree of outlierness among 4 single graph models (DBLP 1 to 4) and for multiple graph models, represented by the most dissimilar models (Ensemble 2 to 4). We average the rankings of 10 iterations and transform the final ranking into bins. There are



(a) DBLP with ConOut using Jaccard similarity measure



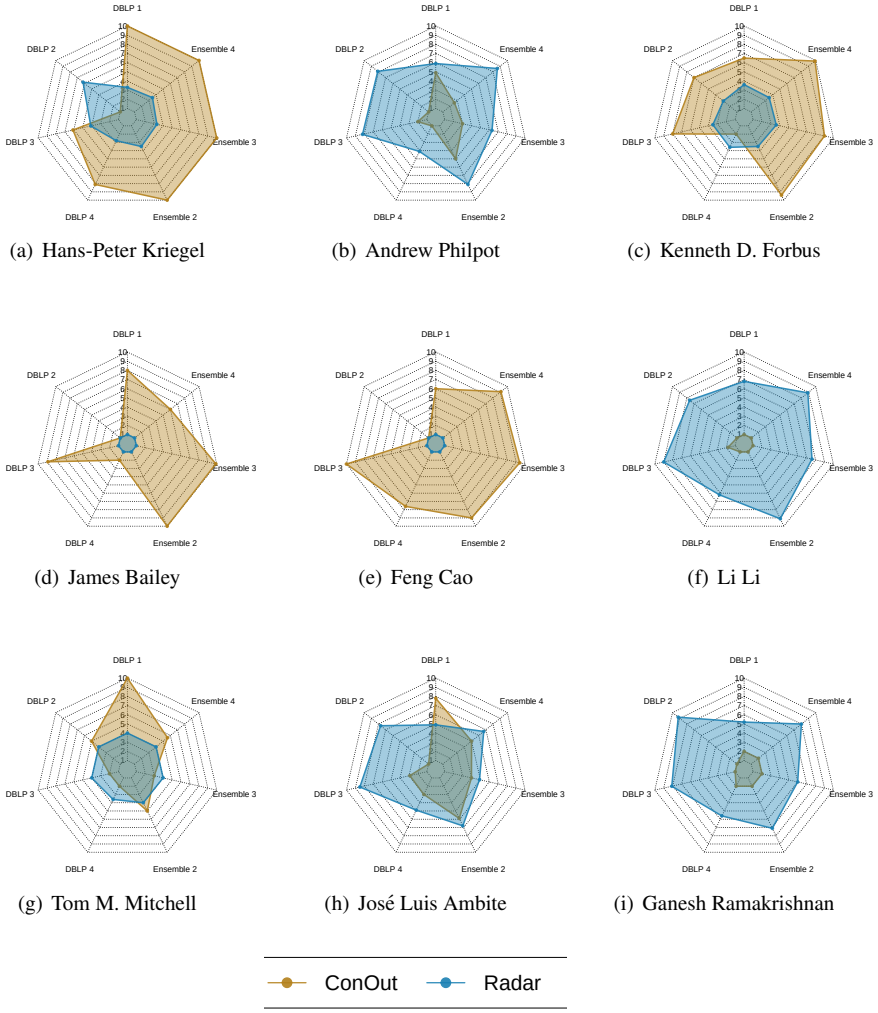
(b) DBLP with ConOut using DeltaCon similarity measure

Fig. 6. ROC AUC on single and multiple graph models on DBLP dataset as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.

10 bins, ranging from 10 (outlier) to 1 (inlier). The bins compress the rankings of outlierness and they increase by a factor of 2 starting from bin 10 covering ranks from 1 to 5, bin 9 covering ranks 6 to 15, bin 8 covering ranks 16 to 35 etc.

Hans-Peter Kriegel is consistently the most prominent outlier node according to ConOut in all DBLP models, except for model 2, in which Radar gives his largest score of outlierness among all models. His pattern is shown in Figure 7(a), where he tends to be an outlier in almost all scenarios according to ConOut.

The same pattern is also present for Feng Cao's and Kenneth D. Forbus's plots, Figures 7(e) and 7(c) respectively. Feng is a top outlier in the DBLP 3 model according to ConOut and exhibits a high degree of outlierness in other single and multiple graph models, but is not a prominent outlier in the DBLP 2 model. Kenneth is not among top 5 of any single graph model, but when we combine multiple graph models, he become top outlier



(a) Hans-Peter Kriegel (b) Andrew Philpot (c) Kenneth D. Forbus

(d) James Bailey (e) Feng Cao (f) Li Li

(g) Tom M. Mitchell (h) José Luis Ambite (i) Ganesh Ramakrishnan

Fig. 7. These plots depict the degree of outlierness for some natural outliers in the DBLP dataset using ConOut and Radar algorithms. The rankings were averaged over 10 iterations and multiple graph models comprise the most dissimilar graphs according to Jaccard. We transform the final ranking into bins, ranging from 10 to 1, where 10 refers to top outlier scores and 1 to inliers scores. We increase the bins range by a factor of 2, starting from bin 10 covering rankings [1 - 5] and ending on bin 1 covering rankings [2556 - Maximum]. Ensembles 2 to 4 correspond to multiple graph models.

Table 6. Average ROC AUC values on Citation Network data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	**0.590**				**0.659**			
2	0.580	0.577	0.545	0.545	0.658	**0.659**	**0.659**	0.658
3	**0.673**	**0.691**	0.570	**0.608**	0.658	**0.659**	**0.659**	**0.659**
4	0.632	0.642	0.517	0.526	**0.659**	**0.659**	**0.659**	**0.659**
	DeltaCon Similarity							
1	**0.590**				**0.659**			
2	**0.646**	**0.723**	0.581	0.581	0.658	0.658	0.658	0.658
3	0.610	0.662	0.510	0.557	**0.659**	**0.659**	**0.659**	**0.659**
4	0.632	0.642	0.517	0.526	**0.659**	**0.659**	**0.659**	**0.659**

according to ConOut. Radar labels Feng Cao as inlier in each model and gives a slightly higher degree of outlierness to Kenneth D. Forbus.

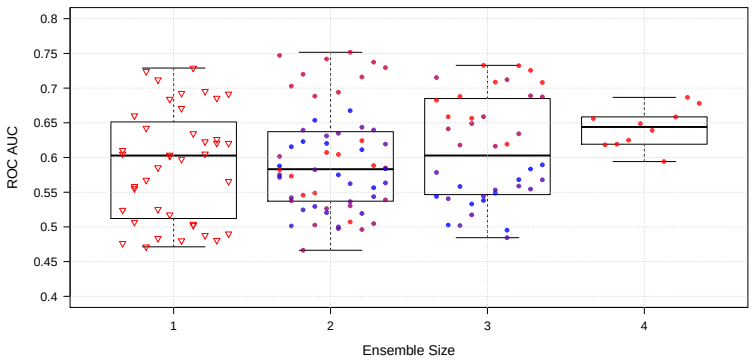
James Bailey and Feng Cao have opossite behavior when compared to Li Li and Ganesh Ramakrishnan regarding the algorithm used to output their degree of outlierness. On the first two, ConOut labels them as outliers and Radar as inliers. On the other hand, Li Li and Ganesh are outliers in Radar's perspective and inliers in ConOut's perspective.

José Luis Ambite and Tom M. Mitchell are clear examples of nodes that are outliers only in a single graph model (DBLP 1) according to ConOut. When we combine dissimilar models, they do not appear unusual anymore. Radar seems very consistent on these two cases, specially on José Luis Ambite, where it outputs high degree of outlierness on models 2 and 3, but slightly reduce his score when we combine multiple graph models.

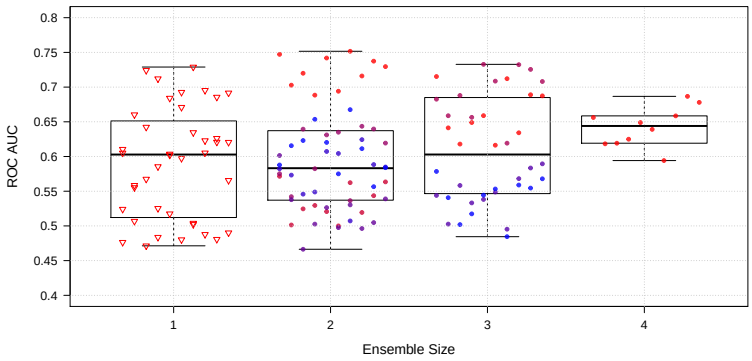
5.3. Citation Network Data

Table 6 shows the average ROC AUC results for ConOut and Radar on single and multiple graph models. We again see the robustness of the Radar algorithm towards different models to represent the same data. DeltaCon similarity measure selects better dissimilar models to combine their outputs than Jaccard on Citation Network when selecting two graph models. With Jaccard best results are achieved when selecting three models. The highest value using multiple graph models is 0.723 average ROC AUC combining 2 most dissimilar models according to Jaccard using ConOut algorithm and combine by maximum score.

Figure 8 shows the results when we apply ConOut to single graph models (ensemble size 1) and to multiple graph models, using borda as combination function. We repeat the same procedure to generate Figure 6, where the colors of the points are set as for the experiments on synthetic data (blue: combination of more similar graph models, red: less similar). Each column within each boxplot relates to one of the 10 dataset variants (different outliers). In Figures 8(a) and 8(b) the boxplots shows the benefits of using multiple graph models for outlier detection task. On Citation Network, the red dots are clearly on top of each experiment, which holds our claim towards selecting more diverse



(a) Citation Network with ConOut using Jaccard similarity measure



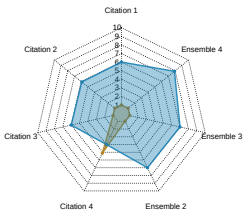
(b) Citation Network with ConOut using DeltaCon similarity measure

Fig. 8. ROC AUC on single and multiple graph models on Citation Network dataset as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.

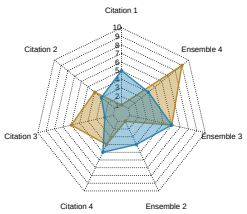
graphs to combine their outputs. Even though when selecting ensemble size equal to 2 in Figure 8(a) the red dots are in the middle of the boxplot, on the top of each experiment are the second most dissimilar models. This effect is shown in Table 6, where Jaccard best results are selecting ensemble size 3.

Together with true outliers ('plagiarized' papers) there are some papers that consistently get high outlier scores. Figure 9 measure the degree of outlierness among 4 single graph models (Citation Network 1 to 4) and for multiple graph models, represented by the most dissimilar models (Ensembles 2 to 4). We average the rankings of 10 iterations and transform the final ranking into bins. There are 10 bins, ranging from 10 (outlier) to 1 (inlier).

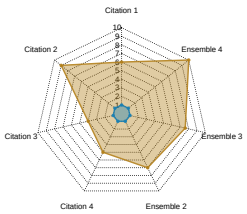
The behavior present in Figures 9(c) and 9(d) is quite similar. Radar labels both as true inliers and ConOut shows a high degree of outlierness on ensemble size 4. Papers



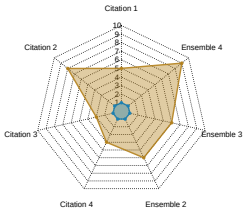
(a) Coloring Terms to Control Equational Reasoning (Journal of Automated Reasoning, 1997)



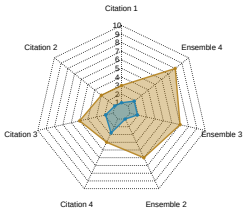
(b) On Broadcast Disk Paging (SIAM Journal on Computing, 2000)



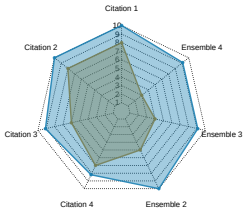
(c) Discriminative Learning for Dynamic State Prediction (IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009)



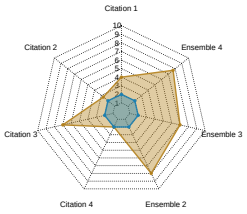
(d) An Approximate and Efficient Method for Optimal Rotation Alignment of 3D Models (IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007)



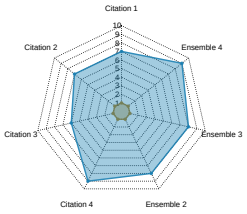
(e) Pairwise Data Clustering by Deterministic Annealing (IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997)



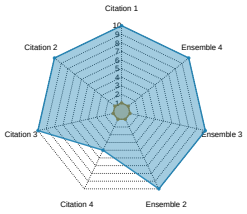
(f) A Class of Inverse Dominant Problems under Weighted l_∞ Norm and an Improved Complexity Bound for Radziś Algorithm (Journal of Global Optimization, 2006)



(g) Motion Estimation with Quadtree Splines (IEEE Transactions on Pattern Analysis and Machine Intelligence, 1996)



(h) Extracting guarantees from chaos (Communications of the ACM, 2003)



(i) A Logic-Based Transformation System (IEEE Transactions on Knowledge and Data Engineering, 1998)

Fig. 9. These plots depict the degree of outlierness for some natural outliers in the Citation Network dataset using ConOut and Radar algorithms. The rankings were averaged over 10 iterations and multiple graph models comprise the most dissimilar graphs according to Jaccard. We transform the final ranking into bins, ranging from 10 to 1, where 10 refers to top outlier scores and 1 to inliers scores. We increase the bins range by a factor of 2, starting from bin 10 covering rankings [1 - 5] and ending on bin 1 covering rankings [2556 - Maximum]. Ensembles 2 to 4 correspond to multiple graph models.

Table 7. Average ROC AUC values on Facebook data experiments when we select the most dissimilar models to combine in each iteration. For single graph models (ensemble size 1), we take the average of all possibilities.

Ens. Size	ConOut				Radar			
	Mean	Max	Median	Borda	Mean	Max	Median	Borda
	Jaccard Similarity							
1	0.919				0.678			
2	0.934	0.919	0.931	0.931	0.686	0.686	**0.685**	**0.685**
3	**0.974**	**0.959**	**0.951**	**0.963**	**0.687**	**0.688**	0.674	0.684
	DeltaCon Similarity							
1	0.919				0.678			
2	0.934	0.919	0.931	0.931	0.686	0.686	**0.685**	**0.685**
3	**0.974**	**0.959**	**0.951**	**0.963**	**0.687**	**0.688**	0.674	0.684

in Figures 9(e) and 9(g) have their degree of outlierness increased only using multiple models, as they are inliers in a single model approach using ConOut. In Figure 9(b), both algorithms have different results considering different graph models. It is only considered an outlier when using the combination of outputs of all graph models with the ConOut algorithm.

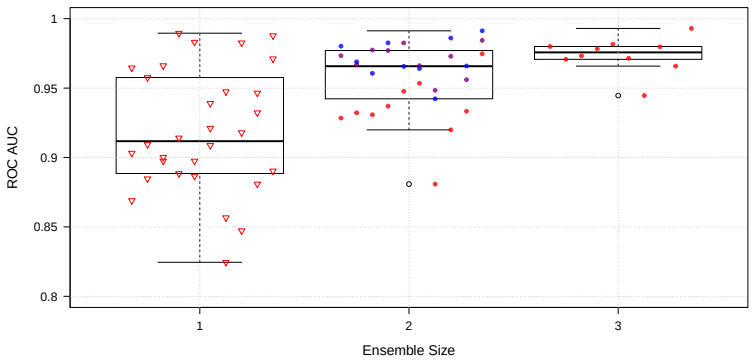
With the papers in Figures 9(a), 9(h) and 9(i) we have inliers according to ConOut and top outliers according to Radar. We see, especially with the paper "A Logic-Based Transformation System", a high degree of outlierness defined by Radar algorithm on single and multiple models. The paper in Figure 9(f) is labeled as outlier by ConOut and Radar.

5.4. Facebook Data

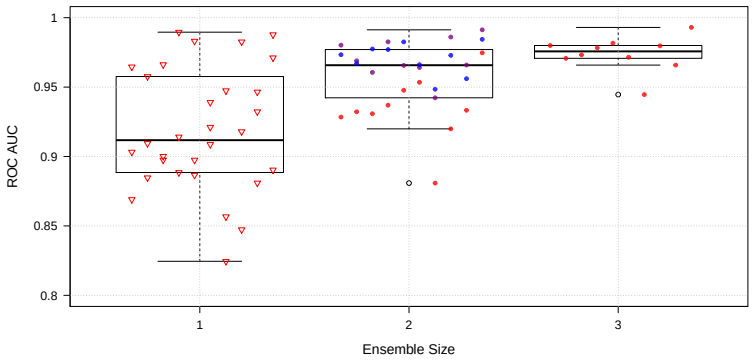
Table 7 shows the average ROC AUC results for ConOut and Radar on single and multiple graph models. Again, all results shows that using multiple graph models improves the outlier detection performance. ConOut shows superior performance compared to Radar, reaching average ROC AUC of 0.974 in the best scenario combining all graph models outputs, in comparison to 0.919 when using only single graphs to represent the data. The best combination technique in this experiment is taking the average score to represent the final degree of outlierness.

Figure 10 shows the ROC AUC results for ConOut on single and multiple graph models, using mean as consensus function. The results are over 10 variants and the columns inside each boxplot show the results for each iteration. Even though the experiments suggest the selection of as many models you have to represent the data, in this specific scenario, combining 2 dissimilar models have lower performance than combining similar models.

Findings on DBLP and Citation Network data can also be applied here. We observe on Facebook data that using multiple graph models improves the outlier detection performance.



(a) Facebook with ConOut using Jaccard similarity measure



(b) Facebook with ConOut using DeltaCon similarity measure

Fig. 10. ROC AUC on single and multiple graph models on Facebook dataset as we vary the ensemble size (number of combined graph models, where 1 represents single graph models). The colors of the points reflect the (average) similarity of the graph models selected for combination: blue (more similar) to red (less similar). Each column of points inside each boxplot presents the results for an independent subset of graphs.

6. Conclusion

Outlier detection is a subjective and unsupervised task that demands good knowledge and understanding of the data. Using a single graph model of relation-rich datasets may only model some aspects of the data, thus not making proper use of potential information. Using multiple graph models may capture more and complementary information. We therefore suggest, based on our findings, to explore real world data using multiple graph models that are as complementary as possible.

In a practical application, a data analyst is interested in certain entities that lend themselves as a set of nodes in a graph representation while several attributes or inter-relational connections may be represented as edges between nodes. Instead of looking for the one and only, best-ever graph representation of some given raw data, the data analyst should

therefore generate multiple graph models describing different aspects of the raw data, capturing a large variety of characteristics, or putting different emphasis on certain characteristics. That is, the graphs may differ both quantitatively (how dense they are) and qualitatively (which relationships are expressed in the graph structure). These multiple graph models aim to materialize the various perspectives that the analyst wants to highlight, that is, they should cover the problem scenario as well as possible and in as many different ways as suitable.

Clearly, many questions remain open. We focused in this study purely on the aspect of the impact of multiple graph models for a given dataset. We evaluated this impact using two different outlier detection algorithms, four combination functions, and two similarity measures on synthetic and real world data. For a practical application, various aspects will have strong influence on the achievable quality, for example the algorithm used to detect outliers on the individual graphs and the method used to combine the individual results (as we have seen in this evaluation). However based on our study we can maintain the recommendation to consider several different graph representations in any case.

Acknowledgments. This work was partially supported by CAPES - Brazil, Fapemig, CNPq, and by projects InWeb, MASWeb, EUBra-BIGSEA (H2020-EU.2.1.1 690116, Brazil/MCTI/RNP GA-000650/04), INCT-Cyber, and Atmosphere (H2020-EU 777154, Brazil/MCTI/RNP 51119).

References

1. Akoglu, L., McGlohon, M., Faloutsos, C.: oddball: Spotting anomalies in weighted graphs. In: Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21-24, 2010. Proceedings. Part II. pp. 410–421 (2010), https://doi.org/10.1007/978-3-642-13672-6_40
2. Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data Min. Knowl. Discov. 29(3), 626–688 (2015)
3. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
4. de Borda, J.C.: Mémoire sur les élections au scrutin. Histoire de l'Académie Royale des Sciences (1784)
5. Campos, G.O., Zimek, A., Sander, J., Campello, R.J.G.B., Micenková, B., Schubert, E., Assent, I., Houle, M.E.: On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. Data Mining and Knowledge Discovery 30, 891–927 (2016)
6. Campos, G.O., Meira Jr., W., Zimek, A.: Outlier detection in graphs: On the impact of multiple graph models. In: Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, WIMS 2018, Novi Sad, Serbia, June 25-27, 2018. pp. 21:1–21:12 (2018), http://doi.acm.org/10.1145/3227609.3227646
7. Chakrabarti, D.: Autopart: Parameter-free graph partitioning and outlier detection. In: Knowledge Discovery in Databases: PKDD 2004, 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, September 20-24, 2004, Proceedings. pp. 112–124 (2004), https://doi.org/10.1007/978-3-540-30116-5_13
8. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: A survey. IEEE Transactions on Knowledge and Data Engineering 24(5), 823–839 (2012)
9. Erdős, P., Rényi, A.: On random graphs i. Publicationes Mathematicae (Debrecen) 6, 290–297 (1959)
10. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: Knowledge discovery and data mining: Towards a unifying framework. In: Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR. pp. 82–88 (1996)

11. Gao, J., Tan, P.N.: Converting output scores from outlier detection algorithms into probability estimates. In: Proceedings of the 6th IEEE International Conference on Data Mining (ICDM), Hong Kong, China. pp. 212–221 (2006)
12. Gao, J., Liang, F., Fan, W., Wang, C., Sun, Y., Han, J.: On community outliers and their efficient detection in information networks. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010. pp. 813–822 (2010)
13. Henderson, K., Gallagher, B., Li, L., Akoglu, L., Eliassi-Rad, T., Tong, H., Faloutsos, C.: It's who you know: graph mining using recursive structural features. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011. pp. 663–671 (2011), http://doi.acm.org/10.1145/2020408.2020512
14. Jaccard, P.: Distribution de la florine alpine dans la bassin de dranses et dans quelques regiones voisines. Bulletin de la Societe Vaudoise des Sciences Naturelles 37, 241–272 (1901)
15. Kirner, E., Schubert, E., Zimek, A.: Good and bad neighborhood approximations for outlier detection ensembles. In: Proceedings of the 10th International Conference on Similarity Search and Applications (SISAP), Munich, Germany. pp. 173–187 (2017)
16. Koutra, D., Ke, T., Kang, U., Chau, D.H., Pao, H.K., Faloutsos, C.: Unifying guilt-by-association approaches: Theorems and fast algorithms. In: Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011, Proceedings, Part II. pp. 245–260 (2011), https://doi.org/10.1007/978-3-642-23783-6_16
17. Koutra, D., Shah, N., Vogelstein, J.T., Gallagher, B., Faloutsos, C.: Deltacon: Principled massive-graph similarity function with attribution. TKDD 10(3), 28:1–28:43 (2016), http://doi.acm.org/10.1145/2824443
18. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Outlier detection in axis-parallel subspaces of high dimensional data. In: Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Bangkok, Thailand. pp. 831–838 (2009)
19. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Interpreting and unifying outlier scores. In: Proceedings of the 11th SIAM International Conference on Data Mining (SDM), Mesa, AZ. pp. 13–24 (2011)
20. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: Proceedings of the 11th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL. pp. 157–166 (2005)
21. Leskovec, J., Krevl, A.: SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data (Jun 2014)
22. Li, J., Dani, H., Hu, X., Liu, H.: Radar: Residual analysis for anomaly detection in attributed networks. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017. pp. 2152–2158 (2017), https://doi.org/10.24963/ijcai.2017/299
23. Liu, C., Yan, X., Yu, H., Han, J., Yu, P.S.: Mining behavior graphs for "backtrace" of non-crashing bugs. In: Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005. pp. 286–297 (2005), https://doi.org/10.1137/1.9781611972757.26
24. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD) 6(1), 3:1–39 (2012)
25. Micenková, B., McWilliams, B., Assent, I.: Learning outlier ensembles: The best of both worlds–supervised and unsupervised. In: Workshop on Outlier Detection and Description under Data Diversity (ODD2), held in conjunction with the 20th ACM International Conference on Knowledge Discovery and Data Mining, New York, NY. pp. 51–54 (2014)

26. Müller, E., Schiffer, M., Seidl, T.: Adaptive outlierness for subspace outlier ranking. In: Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM), Toronto, ON, Canada. pp. 1629–1632 (2010)
27. Müller, E., Sánchez, P.I., Mülle, Y., Böhm, K.: Ranking outlier nodes in subspaces of attributed graphs. In: Workshops Proceedings of the 29th IEEE International Conference on Data Engineering, ICDE 2013, Brisbane, Australia, April 8-12, 2013. pp. 216–222 (2013)
28. Nguyen, H.V., Ang, H.H., Gopalkrishnan, V.: Mining outliers with ensemble of heterogeneous detectors on random subspaces. In: Proceedings of the 15th International Conference on Database Systems for Advanced Applications (DASFAA), Tsukuba, Japan. pp. 368–383 (2010)
29. Noble, C.C., Cook, D.J.: Graph-based anomaly detection. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 24 - 27, 2003. pp. 631–636 (2003)
30. Perozzi, B., Akoglu, L., Sánchez, P.I., Müller, E.: Focused clustering and outlier detection in large attributed graphs. In: The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014. pp. 1346–1355 (2014)
31. Porter, M.F.: An algorithm for suffix stripping. Program 14(3), 130–137 (1980)
32. Rayana, S., Akoglu, L.: Less is more: Building selective anomaly ensembles. ACM Transactions on Knowledge Discovery from Data (TKDD) 10(4), 42:1–42:33 (2016)
33. Rayana, S., Zhong, W., Akoglu, L.: Sequential ensemble learning for outlier detection: A bias-variance perspective. In: Proceedings of the 16th IEEE International Conference on Data Mining (ICDM), Barcelona, Spain. pp. 1167–1172 (2016)
34. Rotabi, R., Kamath, K., Kleinberg, J.M., Sharma, A.: Detecting strong ties using network motifs. In: Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017. pp. 983–992 (2017), http://doi.acm.org/10.1145/3041021.3055139
35. Sánchez, P.I., Müller, E., Irmler, O., Böhm, K.: Local context selection for outlier ranking in graphs with multiple numeric node attributes. In: Conference on Scientific and Statistical Database Management, SSDBM '14, Aalborg, Denmark, June 30 - July 02, 2014. pp. 16:1–16:12 (2014)
36. Sánchez, P.I., Müller, E., Laforet, F., Keller, F., Böhm, K.: Statistical selection of congruent subspaces for mining attributed graphs. In: 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, December 7-10, 2013. pp. 647–656 (2013)
37. Schubert, E., Wojdanowski, R., Zimek, A., Kriegel, H.P.: On evaluation of outlier rankings and outlier scores. In: Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA. pp. 1047–1058 (2012)
38. Schubert, E., Zimek, A., Kriegel, H.P.: Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection. Data Mining and Knowledge Discovery 28(1), 190–237 (2014)
39. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008. pp. 990–998 (2008), http://doi.acm.org/10.1145/1401890.1402008
40. Tong, H., Lin, C.: Non-negative residual matrix factorization with application to graph anomaly detection. In: Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA. pp. 143–153 (2011), https://doi.org/10.1137/1.9781611972818.13
41. Xu, X., Yuruk, N., Feng, Z., Schweiger, T.A.J.: SCAN: a structural clustering algorithm for networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007. pp. 824–833 (2007), http://doi.acm.org/10.1145/1281192.1281280

42. Zimek, A., Campello, R.J.G.B., Sander, J.: Ensembles for unsupervised outlier detection: Challenges and research questions. ACM SIGKDD Explorations 15(1), 11–22 (2013)
43. Zimek, A., Campello, R.J.G.B., Sander, J.: Data perturbation for outlier detection ensembles. In: Proceedings of the 26th International Conference on Scientific and Statistical Database Management (SSDBM), Aalborg, Denmark. pp. 13:1–12 (2014)
44. Zimek, A., Gaudet, M., Campello, R.J.G.B., Sander, J.: Subsampling for efficient and effective unsupervised outlier detection ensembles. In: Proceedings of the 19th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Chicago, IL. pp. 428–436 (2013)
45. Zimek, A., Schubert, E., Kriegel, H.P.: A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analysis and Data Mining 5(5), 363–387 (2012)

Guilherme Oliveira Campos is currently a PhD student in Computer Science at Federal University of Minas Gerais (UFMG). He holds master-level degree in Computer Science at University of São Paulo (USP). His research interests include outlier detection, ensemble techniques for unsupervised learning and anomaly detection in graphs.

Edré Moreira is currently a PhD student in Computer Science at Federal University of Minas Gerais (UFMG). He holds master-level degree in Computer Science at Federal University of Minas Gerais. His research interests include unsupervised learning and anomaly detection in graphs.

Wagner Meira Jr. obtained his PhD in Computer Science from the University of Rochester in 1997 and is Full Professor at the Computer Science Department at Universidade Federal de Minas Gerais, Brazil. He has published more than 200 papers in top venues and is co-author of the book Data Mining and Analysis - Fundamental Concepts and Algorithms published by Cambridge University Press in 2014. His research focuses on scalability and efficiency of large scale parallel and distributed systems, from massively parallel to Internet-based platforms, and on data mining algorithms, their parallelization, and application to areas such as information retrieval, bioinformatics, and e-governance.

Arthur Zimek is Associate Professor in the Department of Mathematics and Computer Science (IMADA) at University of Southern Denmark (SDU), in Odense, Denmark and Head of the Section "Data Science and Statistics". He joined SDU in 2016 after previous positions at Ludwig-Maximilians-University Munich (LMU), Germany, Technical University Vienna, Austria, and University of Alberta, Edmonton, Canada. Arthur holds master-level degrees in bioinformatics, philosophy, and theology, involving studies at universities in Germany (TUM, HfPh, LMU Munich, and JGU Mainz) as well as Austria (LFU Innsbruck). His research interests include ensemble techniques for unsupervised learning, clustering, outlier detection, and high dimensional data, developing data mining methods as well as evaluation methodology. He published more than 80 papers at peer reviewed international conferences and in international journals. He co-organized several workshops and mini-symposia at SDM, KDD, and Shonan and served as workshop co-chair for SDM 2017.

Received: October 10, 2018; Accepted: May 1, 2019.

Computer Science and Information Systems 16(2):597–614 https://doi.org/10.2298/CSIS181001011L

How Much Topological Structure Is Preserved by Graph Embeddings? ⋆

Xin Liu[1], Chenyi Zhuang[1], Tsuyoshi Murata[2],
Kyoung-Sook Kim[1], and Natthawut Kertkeidkachorn[1]

[1] Artificial Intelligence Research Center
National Institute of Advanced Industrial Science and Technology
AIST Waterfront ANNEX, 2-4-7 Aomi, Koto-ku, Tokyo 135-0064 Japan
{xin.liu, zhuang.chenyi, ks.kim, n.kertkeidkachorn}@aist.go.jp
[2] Department of Computer Science
Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552 Japan
murata@c.titech.ac.jp

Abstract. Graph embedding aims at learning representations of nodes in a low dimensional vector space. Good embeddings should preserve the graph topological structure. To study how much such structure can be preserved, we propose evaluation methods from four aspects: 1) How well the graph can be reconstructed based on the embeddings, 2) The divergence of the original link distribution and the embedding-derived distribution, 3) The consistency of communities discovered from the graph and embeddings, and 4) To what extent we can employ embeddings to facilitate link prediction. We find that it is insufficient to rely on the embeddings to reconstruct the original graph, to discover communities, and to predict links at a high precision. Thus, the embeddings by the state-of-the-art approaches can only preserve part of the topological structure.

Keywords: graph embedding, network representation learning, graph reconstruction, dimension reduction, graph mining.

1. Introduction

Graphs (also known as networks) are used in many branches of science as a way to represent the patterns of connections between the components of complex systems [48]. Recently, there has been a surge of interest in *graph embedding* that learns low-dimensional vector representations, or *embeddings*, for nodes to encode their structural information in the original graph [23,3,20,75]. After the embeddings are learned, graph analysis can be easily and efficiently carried out by applying off-the-shelf vector-based machine learning algorithms [59,58,68,6,78,21,56].

It is believed that the topological structure information should, to some extent, be preserved by the embeddings that are obtained by the state-of-the-art approaches. But how well is it preserved? This question is not yet investigated and this paper intends to answer them. In this paper, we propose four evaluation methods to evaluate the amount of information preserved by the embeddings. First, we investigate how well the graph

⋆ This paper is an extension of the conference version [40].

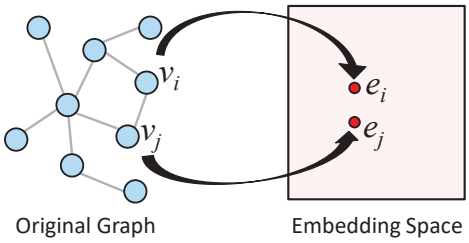


Fig. 1. An illustration of graph embedding.

can be reconstructed by the embeddings. Secondly, we study the divergence between the link distribution in the graph and the distribution derived from the embeddings. Thirdly, we focus on the difference between the communities discovered from the graph and the embeddings. Finally, we examine the effectiveness of embeddings for facilitating link prediction. We found that the current graph embedding approaches can only preserve part of the topological structure. It is insufficient to rely on the embeddings to reconstruct the original graph, to discover communities, and to predict links at a high precision.

The rest of the paper is organized as follows. Section 2 presents the definition of graph embedding. Section 3 proposes our methods in detail. Section 4 reports the experiment results for different graph embedding techniques based on the proposed evaluation methods. Section 5 surveys related work. Finally, Section 6 gives our conclusion.

2. Preliminaries and Definition of Graph Embedding

This section gives definitions of graph embedding. We begin with the symbols that will be used. Let us consider a simple graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_i \mid i = 1, \cdots, n\}$ is the node set, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. We simply suppose the edge weight is uniformly 1. The adjacency matrix of $\mathbf{G}$ is denoted as $\mathbf{A}$, with elements

$$A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in \mathcal{E}; \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$k_i = \sum_{j=1}^{n} A_{ij}$ is the degree of v_i.

Definition: Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, graph embedding is a mapping ϕ: $v_i \mapsto \boldsymbol{e_i} \in \mathbb{R}^d$ for $\forall i = 1, \ldots, n$, such that $d \ll n$ and the embeddings maximally preserve the structure of graph.

The most basic structure that should be preserved is the topological structure. That is, if there is a link between v_i and v_j, the corresponding embeddings $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$ should be close to each other in the low dimension vector space, as shown in Figure 1.

3. Evaluating How Much Structure Are Preserved

In this section we propose four evaluation methods for studying how much graph structure are preserved by the embeddings. Our methods are carried out from four aspects: 1) graph

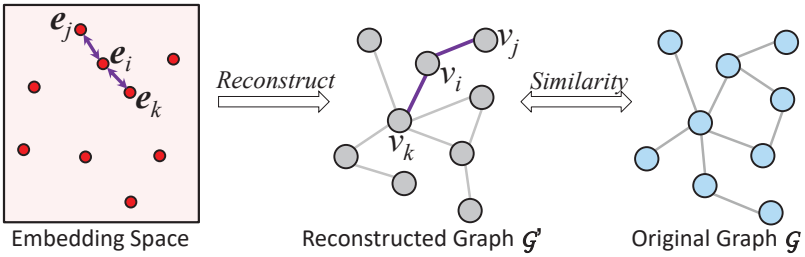


Fig. 2. The two steps for the graph similarity based evaluation.

reconstruction based on the embeddings, 2) the divergence of the original link distribution and the embedding-derived distribution, 3) the consistency of communities discovered from the graph and embeddings, and 4) link prediction based on the embeddings. We will present them in the following.

3.1. Graph Reconstruction based on Embeddings

To evaluate how much topological structure information is preserved by the embeddings, we can use the embeddings to reconstruct a graph and examine the difference between the reconstructed graph and the original one. Following this idea, we propose an evaluation method based on the similarity of the two graphs. Our method contains two steps as illustrated in Figure 2. The first step is reconstructing the graph based on the embedding. The second step is calculating the similarity between the reconstructed graph and the original graph.

Reconstructing the Graph Given the embeddings $\{\boldsymbol{e_i} \mid i = 1, \cdots, n\}$, the similarity function for a pair of embeddings $\textsc{Sim}(\boldsymbol{e_i}, \boldsymbol{e_j}) : (\boldsymbol{e_i}, \boldsymbol{e_j}) \rightarrow \mathbb{R}$, and the node degree sequence $\{k_i \mid i = 1, \cdots, n\}$ of the original graph, we take the following procedure to obtain the reconstructed graph $\mathcal{G}'$.

1. $\mathcal{G}'$ keeps the same node set $\{v_i \mid i = 1, \cdots, n\}$ as $\mathcal{G}$.
2. For each v_i whose degree is k_i in $\mathcal{G}$, create k_i links connecting v_i and the nodes whose corresponding embeddings are among the k_i most similar embeddings to $\boldsymbol{e_i}$, with the weight of each link equal to 0.5.

$\textsc{Sim}(\boldsymbol{e_i}, \boldsymbol{e_j})$ quantifies the similarity of $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$ in the embedding space, and is dependent on the approach for generating the embeddings. For example, if an approach expresses the similarity by dot product, the similarity function would be based on dot product.

Note that for each created link we attach a weight of 0.5. This is because the link creation is a mutual process, i.e., for v_i we create a link to v_j, and for v_j we may create another link to v_i. As a result, $\mathcal{G}'$ keeps the same number of weights as $\mathcal{G}$.

Also note that $\mathcal{G}'$ can be exactly the same as $\mathcal{G}$ under the condition that for each node v_i, the k_i most similar embeddings of $\boldsymbol{e_i}$ exactly correspond to the neighbor nodes of v_i. Therefore, if the embeddings are good enough we can perfectly reconstruct the graph.

Evaluating the Graph Similarity Good embeddings that well preserve the topological structure will result in that the reconstructed graph $\mathcal{G}'$ is similar to the original graph $\mathcal{G}$. Thus, we can evaluate the amount of preserved information by calculating the similarity between $\mathcal{G}$ and $\mathcal{G}'$. Specifically, we use DELTACON [32,31] as a metric to measure the similarity. DELTACON is scalable to large graphs and obeys the following axioms

- *Identity Property*: $\text{DELTACON}(\mathcal{G}, \mathcal{G}') = 1$ iff $\mathcal{G} = \mathcal{G}'$.
- *Symmetric Property*: $\text{DELTACON}(\mathcal{G}, \mathcal{G}') = \text{DELTACON}(\mathcal{G}', \mathcal{G})$.
- *Zero Property*: $\text{DELTACON}(\mathcal{G}, \mathcal{G}') \to 0$ for $n \to \infty$, where $\mathcal{G}$ is the complete graph, and $\mathcal{G}'$ is the empty graph (i.e., the edge sets are complementary).

DELTACON essentially measures the differences in the corresponding node's affinity of $\mathcal{G}$ and $\mathcal{G}'$, and thus it is based on global structure of the graphs. Specifically, the calculation of $\text{DELTACON}(\mathcal{G}, \mathcal{G}')$ contains three steps. First, we calculate the node affinity matrices $\mathbf{S}$ and $\mathbf{S}'$ for $\mathcal{G}$ and $\mathcal{G}'$, respectively. The node affinity matrix $\mathbf{S}$ can be expressed as

$$\mathbf{S} = (\mathbf{I} + \epsilon^2 \mathbf{D} - \epsilon \mathbf{A})^{-1}, \tag{2}$$

where ϵ is a positive constant encoding the influence between neighbors in $\mathcal{G}$, and $\mathbf{D}$ is the degree diagonal matrix, with elements

$$D_{ij} = \begin{cases} k_i & \text{if } i = j; \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The element S_{ij} indicate the affinity (influence) of node v_i to v_j in $\mathcal{G}$. Similarly, we calculate the node affinity matrix $\mathbf{S}'$ for $\mathcal{G}'$. Secondly, we calculate the root Euclidean distance between $\mathbf{S}$ and $\mathbf{S}'$.

$$\text{ROOTED}(\mathbf{S}, \mathbf{S}') = \sqrt{\sum_{i,j} (S_{ij} - S'_{ij})^2} \tag{4}$$

Finally, we have

$$\text{DELTACON}(\mathcal{G}, \mathcal{G}') = \frac{1}{1 + \text{ROOTED}(\mathbf{S}, \mathbf{S}')}. \tag{5}$$

$\text{DELTACON}(\mathcal{G}, \mathcal{G}') \in [0, 1]$. On the one extreme, a score of 0 implies that $\mathcal{G}'$ is totally irrelevant of $\mathcal{G}$, implying that none of the topological structure information is preserved in the embeddings. On the other extreme, a score of 1 indicates that $\mathcal{G}'$ is a perfect reconstruction of $\mathcal{G}$, implying that the topological structure are 100% preserved in the embeddings. Intermediate scores suggest situations in between the two extremes.

The graph reconstruction procedure requires a quadratic time complexity, since we need to calculate $\text{SIM}(\boldsymbol{e_i}, \boldsymbol{e_j})$ for each pair of embeddings. Given the original graph and the reconstructed graph, the calculation of $\text{DELTACON}(\mathcal{G}, \mathcal{G}')$ needs another quadratic time complexity [32,31]. Therefore, the whole evaluation method requires $\mathcal{O}(n^2)$ complexity.

3.2. Divergence of the Original and Embedding-Derived Link Distributions

Our second method is to evaluate the KL Divergence between the link distribution derived from the embeddings and the empirical distribution observed from the original graph. Given the embeddings $\{\boldsymbol{e_i} \mid i = 1, \cdots, n\}$ and their similarity function $\textsc{Sim}(\boldsymbol{e_i}, \boldsymbol{e_j})$, we can define a link distribution

$$P^{\mathrm{e}}(v_i, v_j) \propto \frac{1}{1 + \exp(-\textsc{Sim}(\boldsymbol{e_i}, \boldsymbol{e_j}))}. \tag{6}$$

The idea is that the more similar $\boldsymbol{e_i}$ and $\boldsymbol{e_j}$ are, the more likely a link will exist between v_i and v_j. This is the link distribution derived from the embeddings. On the other hand, the empirical link distribution observed from the original graph is

$$P^{\mathrm{g}}(v_i, v_j) = \frac{A_{ij}}{\sum_{i<j} A_{ij}}. \tag{7}$$

P^{e} and P^{g} are distributions defined over the space $\mathcal{V} \times \mathcal{V}$. We can employ the KL-divergence [33]

$$\mathrm{KL}(P^{\mathrm{e}}, P^{\mathrm{g}}) = -\sum_{v_i, v_j} P^{\mathrm{e}}(v_i, v_j) \log P^{\mathrm{g}}(v_i, v_j) + \sum_{v_i, v_j} P^{\mathrm{e}}(v_i, v_j) \log P^{\mathrm{e}}(v_i, v_j) \tag{8}$$

to measure the distance between the two distributions. $\mathrm{KL}(P^{\mathrm{e}}, P^{\mathrm{g}})$ approaching 0 indicates that the topological structure are well preserved in the embeddings.

Note that the calculation of P^{g} needs a linear time complexity, while the the calculation of P^{e} needs a quadric time complexity. As a result, the total complexity for this evaluation method is $\mathcal{O}(n^2)$.

3.3. Consistency of Communities Discovered from the Graph and Embeddings

Good embeddings should also preserve the mesoscopic graph structure, i.e., the community structure (clusters). Therefore, the third method for evaluating how well the topological structure is preserved is to measure the consistency of communities discovered from the original graph and from embeddings. Specifically, we employ the Louvain algorithm [1] and the K-Means algorithm [42] to discover the communities from the graph and embeddings, respectively [3]. Then, we estimate the consistency of the communities based on the Normalized Mutual Information (NMI) [17,12] and Adjusted Rand Index (ARI) [27].

Suppose Ω^g and Ω^e are community partitions for the graph and embeddings (node/embedding community label assignments). NMI is an information theoretic measure that calculates the amount of common information between two partitions:

$$\mathrm{NMI}(\Omega^g, \Omega^e) = \frac{-2 \sum_{i=1}^{c^g} \sum_{j=1}^{c^e} n_{ij}^{ge} \log(n_{ij}^{ge} n / n_i^g n_j^e)}{\sum_{i=1}^{c^g} n_i^g \log(n_i^g / n) + \sum_{j=1}^{c^e} n_j^e \log(n_j^e / n),} \tag{9}$$

[3] Note that we never know the true community structure. Hence we choose the most popular and widely accepted algorithms for detecting the communities.

where c^g is number of communities in Ω^g, c^e is number of communities in Ω^e, n is the total number of nodes, n_i^g is the number of nodes in the i-th community of Ω^g, n_j^e is the number of nodes in the j-th community of Ω^e, and n_{ij}^{ge} is the number of nodes that are both in the i-th community of Ω^g and the j-th community of Ω^e. If Ω^g and Ω^e match completely, we have a maximum NMI value of 1.0, whereas if Ω^g and Ω^e are totally independent of one another, we have a minimum NMI value of 0.0.

On the other hand, ARI computes a similarity by considering all pairs of samples and counting pairs that are assigned in the same or different communities in Ω^g and Ω^e. The mathematical definition of ARI is

$$\text{ARI}(\Omega^g, \Omega^e) = \frac{\sum_{i=1}^{c^g}\sum_{j=1}^{c^e}\binom{n_{ij}^{ge}}{2} - \left[\sum_{i=1}^{c^g}\binom{n_i^g}{2}\sum_{j=1}^{c^e}\binom{n_j^e}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_{i=1}^{c^g}\binom{n_i^g}{2} + \sum_{j=1}^{c^e}\binom{n_j^e}{2}\right] - \left[\sum_{i=1}^{c^g}\binom{n_i^g}{2}\sum_{j=1}^{c^e}\binom{n_j^e}{2}\right]/\binom{n}{2}}. \tag{10}$$

The ARI has the maximum value 1 when Ω^g and Ω^e agree perfectly, and its expected value is 0 in the case that Ω^g and Ω^e are totally independent of one another. A larger ARI means a higher agreement between Ω^g and Ω^e.

The Louvain algorithm has time complexity of $\mathcal{O}(n\log n)$, while the K-Means algorithm has time complexity of $\mathcal{O}(nc^e dl)$, where l is the number of iterations for the algorithm to converge. Moreover, the calculation of NMI and ARI has time complexity $\mathcal{O}(n^2 max(c^g, c^e))$ in the worst case. Note that $c^g, c^e, d, l \ll n$ and thus can be ignored. Consequently, the total complexity for this evaluation method is $\mathcal{O}(n^2)$.

3.4. Link Prediction Based on Embeddings

Finally, we evaluate the effectiveness of the embeddings for facilitating link prediction. This is based on the following idea: Suppose embeddings can well preserve the graph topological structure; If we remove a small amount of the topology information of the original graph, the resulting embeddings should still keep the main structure of the graph somehow; Therefore, we can use the embedding to facilitate the recovery of some of the removed information, i.e., link prediction.

Specifically, given a graph $\mathcal{G}$ we remove 10% of the links and obtain $\mathcal{G}'$. We test how the embeddings learned from $\mathcal{G}'$ can help predict the removed links. Suppose we focus on v_i, and (v_i, v_j) is a removed link that we aim to predict. Also note that $\text{SIM}(\boldsymbol{e_i}, \boldsymbol{e_k})$ is the score for predicting (v_i, v_k) for $\forall\ v_k$. Then, given the query (v_i, v_j), we can rank v_j against all other nodes [4] based on the scores. A high rank for v_j indicates that we are able to predict (v_i, v_j) in a positive sense. Finally, we evaluate the performance for all the queries based on Mean Reciprocal Rank (MRR) [53] and HITS@K:

[4] We filter out the nodes v_k that already has a link (v_i, v_k) in $\mathcal{G}'$.

$$\text{MRR} = \frac{1}{|\mathcal{Q}|}\sum_{t=1}^{|\mathcal{Q}|}\frac{1}{\text{Rank}_t} \tag{11}$$

$$\text{HITS@K} = \frac{1}{|\mathcal{Q}|}\sum_{t=1}^{|\mathcal{Q}|}\text{Hits}_t \tag{12}$$

$$\text{Hits}_t = \begin{cases} 1 & \text{if Rank}_t \leq \text{K}; \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

where $\mathcal{Q}$ is the set of queries, Rank_t refers to the rank position of v_j for the t-th query (v_i, v_j). MRR $\in (0, 1]$ and HITS@K $\in [0, 1]$. A maximum value of 1.0 implies we can predict all the links perfectly.

For a query (v_i, v_j), the procedure for ranking v_j against all other nodes requires a time complexity of $\mathcal{O}(n)$. Therefore, the total complexity for this evaluation method is $\mathcal{O}(|\mathcal{Q}|\, n)$.

4. Experiment

In this section, we show the results based on the proposed evaluation methods. We consider the following graph embedding approaches that represent the state-of-the-art.

- GraRep (GRep) [4]: This approach defines a loss function by integrating the transition probabilities. Minimizing this loss function has proven to be equivalent to factorizing a matrix that is related to the k-step transition probability matrix. For each k the factorization produces a sub-embedding. Then it concatenates sub-embeddings on different k as the final embedding solution.
- HOPE [50]: This approach learns embeddings by factorizing the Katz similarity [29] matrix. It uses generalized Singular Value Decomposition algorithm to obtain the embeddings efficiently.
- DeepWalk (DW) [51]: This approach first transforms a graph into a collection of linear sequences of nodes using multiple random walks. It then learns embeddings by applying the Skip-Gram model [46,47], originating from natural language processing, to the node sequence.
- Node2Vec (N2V) [21]: This approach is a variant of DeepWalk. It also samples node sequences and feed them to the Skip-Gram model. Instead of DeepWalk's random search sampling strategy, Node2Vec uses 2nd-order random walks that can bias towards a particular search strategy.
- LINE [57]: This approach learns d-dimensional embeddings in two separate phases. In the first phase, it learns $d/2$ dimensions by BFS-style simulations over immediate neighbors of nodes. In the second phase, it learns the next $d/2$ dimensions by sampling nodes strictly at a 2-hop distance from the source nodes. Finally, it concatenates the embeddings learned at the two phases.
- GRA [41]: This approach learns embeddings by factorizing a Global Resource Allocation similarity matrix that is an extension of the Katz and Resource Allocation similarities [80].

Table 1. Statistics of the datasets: number of nodes $|\mathcal{V}|$; number of edges $|\mathcal{E}|$.

Dataset	$\|\mathcal{V}\|$	$\|\mathcal{E}\|$
Kaggle3059	157	2,474
Kaggle4406	399	3,412
BrazilAir	131	1,003
EuropeAir	399	5,993
USAir	1,190	13,599
Cora	2,708	5,278
Citeseer	3,264	4,551
DBLP	13,184	47,937
WikiPage	2,363	11,596
WikiWord	4,777	92,295
PPI	3,860	37,845
BlogCatalog	10,312	333,983

According to the mechanism of these approaches, the embeddings' similarity fuction can be uniformly expressed as

$$\textsc{Sim}(\boldsymbol{e_i}, \boldsymbol{e_j}) = \boldsymbol{e_i}^\top \boldsymbol{e_j}. \tag{14}$$

In addition, we consider randomly generated embeddings as a baseline. We do not include approaches for supervised graph embedding because they require additional information such as node labels for training [74,61,63,30].

We set the embedding dimension as 120 for all approaches. Moreover, the parameter settings for these approaches are the same as the original literature. Specifically, for DeepWalk and Node2Vec, we set the window size to 10, the walk length to 80, and the number of walks per node to 10. For HOPE and GRA, we set the decay rate to 0.95 divided by the spectral radius of $\mathbf{A}$ and $\mathbf{AD}^{-1}$, respectively. For LINE, we set the number of negative samples to 5. For GraRep, we set the maximum transition step to 6. Lastly, for Node2Vec, we obtain the best in-out and return hyperparameters based on a grid search over $\{0.25, 0.50, 1, 2, 4\}$.

We use a variety of real-world graphs from various domains as the testing datasets. A brief description of them follows.

- Kaggle3059 [8][5], Kaggle4406 [8][3]: Graphs representing the friendship of Facebook users.
- BrazilAir [55][6], EuropeAir [55][7], USAir [55][8]: Graphs representing the air traffics in Brazil, Europe, and the USA, respectively. Nodes correspond to airports and edges denote the existence of commercial flights.
- Cora [72][9], Citeseer [30][7], DBLP [56][10]: Graphs representing the citation relationship of scientific papers.

[5] https://www.kaggle.com/c/learning-social-circles/data
[6] http://www.anac.gov.br/
[7] http://ec.europa.eu/
[8] https://transtats.bts.gov/
[9] https://linqs.soe.ucsc.edu/data/
[10] https://aminer.org/billboard/citation/

Table 2. DELTACON scores for different approaches (the higher the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	0.2911	**0.5687**	0.4673	0.4677	0.5074	0.3830	0.4083
Kaggle4406	0.3780	0.5640	0.5445	0.5831	**0.5947**	0.4658	0.5289
BrazilAir	0.3522	0.4448	0.4624	0.3744	**0.5458**	0.3625	0.3857
EuropeAir	0.3814	**0.4592**	0.4085	0.4125	0.4522	0.4333	0.4542
USAir	0.3532	0.4774	0.4010	0.3830	**0.4862**	0.3898	0.4231
Cora	0.4259	0.4880	0.5408	0.6099	0.5556	0.5516	**0.6146**
Citeseer	0.4280	0.4681	0.5384	0.5730	0.5522	0.5510	**0.6202**
DBLP	0.4001	0.4855	0.4947	0.5255	0.5064	0.5512	**0.5907**
WikiPage	0.4008	0.5280	0.5510	**0.5777**	0.5610	0.5407	0.5564
WikiWord	0.4051	0.4752	0.4746	0.4830	0.5068	0.5022	**0.5168**
PPI	0.3999	0.4753	0.4903	0.4945	0.5052	0.5102	**0.5223**
BlogCatalog	0.3882	0.4579	0.4700	0.4495	0.4794	0.4746	**0.4838**

Table 3. KL-divergence scores for different approaches (the lower the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	1.6519	**1.2567**	1.5330	1.5347	1.5068	1.5229	1.5170
Kaggle4406	3.2092	**2.6315**	3.0915	3.0773	2.9063	3.0967	3.1023
BrazilAir	2.2033	**1.9945**	2.0438	2.0890	2.1268	2.1258	2.1119
EuropeAir	2.6476	**2.4191**	2.5451	2.5748	2.5616	2.5327	2.5126
USAir	4.0158	**3.5418**	3.9185	3.9315	3.8230	3.8590	3.8406
Cora	6.6088	**5.9681**	6.5171	6.5020	6.2449	6.5199	6.5378
Citeseer	7.1230	**6.5599**	7.0393	6.9926	6.7138	7.0588	7.0621
DBLP	7.5638	**6.9417**	7.4929	7.5001	7.2152	7.4774	7.4936
WikiPage	5.5452	**4.9841**	5.4620	5.4642	5.3207	5.3809	5.4388
WikiWord	4.9990	**4.6942**	4.8085	4.8823	4.8139	4.7442	4.7234
PPI	5.3436	**4.9735**	5.2719	5.2673	5.2381	5.1422	5.1680
BlogCatalog	5.2800	**4.8913**	5.0656	5.0847	5.0431	5.0068	4.9814

- WikiPage [61][11]: A graph of webpages in Wikipedia, with edges indicating hyperlinks.
- WikiWord [21][12]: A co-occurrence graph of words appearing in Wikipedia.
- PPI [21][8]: A protein-protein interaction graph for Homo Sapiens.
- BlogCatalog [58,8][13]: A graph of social relationships of the bloggers listed on the BlogCatalog website.

Table 1 summarizes the number of nodes and edges in each dataset. Table 2 lists the DELTACON for graph reconstruction. Table 3 presents the KL-divergence of the original

[11] https://github.com/thunlp/MMDW/tree/master/data/

[12] http://snap.stanford.edu/node2vec/#datasets/

[13] http://socialcomputing.asu.edu/datasets/BlogCatalog3/

Table 4. NMI scores for different approaches (the higher the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	0.0839	0.7595	0.5454	0.7190	0.7272	0.7585	**0.7755**
Kaggle4406	0.1397	0.8792	0.5422	0.8530	0.9014	0.8692	**0.9452**
BrazilAir	0.0713	**0.4893**	0.0992	0.1781	0.3097	0.3660	0.4626
EuropeAir	0.0165	0.5725	0.1471	**0.7049**	0.4602	0.6161	0.6632
USAir	0.0349	0.6077	0.2250	0.5974	0.6149	0.6055	**0.6524**
Cora	0.1919	0.6592	0.4280	0.6218	**0.7452**	0.7099	0.7229
Citeseer	0.5153	0.6108	0.4371	0.7721	**0.8825**	0.8192	0.8272
DBLP	0.0110	0.6315	0.2515	0.5581	**0.7186**	0.6819	0.7072
WikiPage	0.0270	0.6287	0.2488	0.5863	0.6163	0.6323	**0.6827**
WikiWord	0.0033	**0.1833**	0.0346	0.1676	0.0599	0.1661	0.1824
PPI	0.0091	0.4163	0.0866	0.3746	0.3528	0.4002	**0.4275**
BlogCatalog	0.0010	0.5220	0.0088	0.3697	0.3331	0.5221	**0.5478**

Table 5. ARI scores for different approaches (the higher the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	0.0129	**0.6126**	0.3320	0.5528	0.5277	0.5157	0.5352
Kaggle4406	0.0027	0.7543	0.1189	0.6697	0.7784	0.7226	**0.8986**
BrazilAir	0.0236	**0.4083**	0.0329	0.0885	0.1531	0.1881	0.2963
EuropeAir	-0.0002	0.4179	0.0861	**0.5641**	0.1565	0.4477	0.5096
USAir	0.0000	0.3676	-0.0164	0.3529	0.3612	0.3416	**0.5051**
Cora	0.0001	0.2242	0.0107	0.1870	**0.4114**	0.2628	0.2922
Citeseer	0.0002	0.0066	-0.0232	0.1245	**0.3820**	0.1787	0.1966
DBLP	0.0000	0.3995	0.0093	0.2953	**0.5848**	0.4721	0.4910
WikiPage	0.0014	0.4537	0.0180	0.3816	0.4346	0.4539	**0.5435**
WikiWord	0.0004	**0.1481**	0.0183	0.1150	0.0706	0.1101	0.1348
PPI	-0.0001	0.3159	0.0039	0.2829	0.2483	0.3324	**0.3390**
BlogCatalog	-0.0001	0.4430	-0.0120	0.2892	0.3145	0.4524	**0.4681**

and embedding-derived link distributions. Table 4 and Table 5 report the NMI and ARI for the consistency of the communities discovered from graph and embeddings. Table 6 and Table 7 reveal the MRR and HITS@K [14] for link prediction based on embeddings. Based on these results, we have the following observations.

– The DELTACON for evaluating graph reconstruction reveals that GraRep and GRA are more successful in smaller graphs such as Kaggle3059, Kaggle4406, BrazilAir, EuropeAir, and USAir. On the other hand, Node2Vec outperforms the others in 6 larger graphs including Cora, Citeseer, DBLP, WikiWord, PPI, and BlogCatalog, but shows less success in the other graphs (especially in Kaggle3059 and BrazilAir). This

[14] We only review the result for K=10 since we experience similar behaviors for other values of K.

Table 6. MRR scores for different approaches (the higher the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	0.0454	0.5109	0.5414	0.4473	**0.5791**	0.1041	0.1225
Kaggle4406	0.0163	0.4135	0.4270	0.4789	**0.5213**	0.1212	0.1535
BrazilAir	0.0621	0.3682	0.4816	0.1274	**0.4935**	0.0573	0.0845
EuropeAir	0.0194	0.1256	0.1744	0.1216	**0.2281**	0.0207	0.0272
USAir	0.0091	0.2359	0.2883	0.1550	**0.3818**	0.0160	0.0181
Cora	0.0038	0.2587	0.1782	**0.6701**	0.2071	0.1669	0.1727
Citeseer	0.0015	0.2133	0.1945	**0.5825**	0.1999	0.1493	0.1577
DBLP	0.0008	0.1359	0.0880	**0.2809**	0.1072	0.0386	0.0508
WikiPage	0.0031	0.2849	0.2421	**0.4402**	0.2925	0.1289	0.1353
WikiWord	0.0022	0.0485	0.0239	**0.0529**	0.0469	0.0048	0.0061
PPI	0.0019	0.0726	0.0586	**0.1091**	0.0778	0.0045	0.0055
BlogCatalog	0.0013	0.0210	0.0163	0.0063	**0.0220**	0.0011	0.0012

is because that Node2Vec uses two hyperparameters to control the search strategy and this enables it to learn long-term dependencies in larger graphs.

- The KL-divergence for evaluating the divergence of the original and embedding-derived link distribution suggests that GraRep demonstrates the best performance in all of the graphs. A main reason is that GraRep is adapt in separating the embeddings of dissimilar nodes, i.e., putting the embeddings of dissimilar nodes far away from each other. Eq. (6) indicates that in the derived link distribution there is a probability for each pair of nodes, while Eq. (7) implies that in the empirical link distribution only a few pairs of nodes have link probability. Therefore, properly separating the embeddings of dissimilar nodes will help achieve a better KL-divergence score.
- The NMI for evaluating the consistency of the communities discovered from graph and embeddings indicates that Node2Vec achieves good results in Kaggle3059, Kaggle4406, USAir, Cora, Citeseer, WikiPage graphs. A common feature of these graph is that they are unconnected. This means that the graph is naturally separated into several communities, each representing a connected component. Therefore, the community partitions by graph and embeddings can easily reach a relatively high agreement for a unconnected graph, and contribute to the high NMI scores. On the other hand, the ARI scores are more strict on the exact partition of a large connected component into small communities. Hence, the ARI scores are much lower than NMI scores. For example, Node2Vec obtains NMI scores of 0.7229 and 0.8272 on Cora and Citeseer graphs, while the corresponding ARI scores are as low as 0.2922 and 0.1966, respectively.
- The MRR and HITS@10 for evaluating the embedding-based link prediction indicate the similar performance patterns. LINE and GRA outperforms the other approaches by a large margin. For example, GRA achieves passable performance in Kaggle3059, Kaggle4406, and BrazilAir graphs, while LINE delivers an acceptable performance in Cora and Citeseer graphs.
- The performances are graph-dependent. For example, although LINE exhibits good performances in 6 graphs for link prediction, it is far below the other approaches

Table 7. HITS@10 scores for different approaches (the higher the better).

Dataset	Approaches						
	Random	**GRep**	**HOPE**	**LINE**	**GRA**	**DW**	**N2V**
Kaggle3059	0.0907	0.7923	0.8226	0.8065	**0.8367**	0.2359	0.2500
Kaggle4406	0.0278	0.7149	0.7398	0.7339	**0.7865**	0.2573	0.3260
BrazilAir	0.1535	0.5891	0.7574	0.2178	**0.7772**	0.1040	0.1634
EuropeAir	0.0300	0.2508	0.3567	0.2025	**0.4767**	0.0267	0.0392
USAir	0.0129	0.3974	0.4551	0.2471	**0.5882**	0.0287	0.0294
Cora	0.0047	0.4498	0.3438	**0.9290**	0.3835	0.3314	0.3371
Citeseer	0.0000	0.4221	0.3739	**0.8816**	0.3849	0.2807	0.3048
DBLP	0.0005	0.2781	0.1845	**0.5541**	0.2252	0.0782	0.1049
WikiPage	0.0047	0.4664	0.3935	**0.6543**	0.4677	0.2138	0.2259
WikiWord	0.0022	0.0870	0.0369	**0.0954**	0.0912	0.0066	0.0093
PPI	0.0024	0.1312	0.1004	**0.2020**	0.1466	0.0053	0.0074
BlogCatalog	0.0018	0.0393	0.0295	0.0112	**0.0409**	0.0012	0.0013

in another two graphs (BrazilAir and EuropeAir). Similarly, GRA dramatically outperforms the others for community discovery in graphs such as Cora, Citeseer, and DBLP. However, it is less successful in graphs such as BrazilAir and WikiWord.

– The performances are also task-dependent. For example, GraRep consistently outperforms the others in all of the 12 graphs for KL-divergence scores, but it just puts in an average performance in the other three tasks. Similarly, Node2Vec demonstrates acceptable performance in graph reconstruction and community discovery, but it conspicuously fails in link prediction.
– The DELTACON, KL-divergence, NMI, ARI, MRR, and HITS@10 scores all indicate that graph embedding approaches are significantly outperforms the randomly generated embeddings. However, they are far from perfect. For example, the DELTACON scores mostly range between 0.4 and 0.6, but none of the approaches obtains scores closing to 1.0. NMI and ARI scores in graphs such as BrazilAir and WikiWord indicates that the embedding communities are quite different from graph communities. Moreover, the MRR and HITS@10 scores in large graphs such as WikiWord, PPI, and BlogCatalog imply that embeddings are not always trustworthy for link prediction. Therefore, the embeddings preserve only part of the topological structure of the graph. It is insufficient to rely on the embeddings to reconstruct the original graph, to discover communities, and to predict links at a high precision. This fact applies to approaches such as HOPE and LINE that is originally designed to preserve high-order proximity of the graph. One important reason is because of the highly non-linear structure of the graph, which poses a great challenge.

5. Related Work

Recently, the graph embedding problem has attracted a great deal of interest. Researchers have proposed various approaches such as matrix factorization [4,7,50,72] and deep neural networks [63,5,30,22,35,34,64,11,69]. The topics are also varied, including unsuper-

vised graph embedding [4,50,51,57,41,39], supervised graph embedding [74,61,63,41,21,30], community preserving embedding [68,6,78,10], and embedding in graphs of various types, such as the bipartite graphs [19], the heterogeneous graphs [28,7,56,14,18,65,76], the multi-relational graphs [52,45], the signed graphs [9,67,65], the uncertain graphs [24], the incomplete graphs [73], the dynamic graphs [77,36,79,44], the scale-free graphs [15], the hyper-graphs [62], and the attributed graphs (accompanied with attribute information such as categories and texts on nodes or edges, or both) [71,67,36,60,25,26,37].

This paper studies the problem of how well the topological structure information is preserved by the embeddings. One relevant research are [50,63] that use the metric PRECISION@K for measuring the graph reconstruction precision. However, this metric is based on graph reconstruction at the local scale of each node and thus cannot give an trustworthy evaluation. We give an explanation using the illustrations in Figure 3. Figure 3(a) shows a graph with the top node weakly connected to a cluster of 6 nodes. Figure 3(b) and Figure 3(c) are two reconstructed graphs of Figure 3(a). It is obvious that Figure 3(b) is a better reconstruction since the main structure of the original graph are kept. On the other hand, Figure 3(c) is a worse reconstruction, since the top node becomes a member of the cluster and the graph structure has been totally changed. However, the local metric PRECISION@K fails to discriminate the two examples. For example, let us look at node '3' and '4'. The evaluation of these two nodes based on PRECISION@2 can be as high as 1.0 for Figure 3(c) while it is as low as 0.5 for Figure 3(b). On the other hand, the global metric DELTACON can provide an unbiased evaluation of 0.6477 for Figure 3(b) and 0.5584 for Figure 3(c). Moreover, PRECISION@K is computationally expensive, especially for large graphs. To fasten the computation, we usually employ the sampling strategy, but it will cause a serious problem of unstable evaluation. To the best of our knowledge, we are the first to propose the evaluation for graph reconstruction at the global scale of graphs.

There are research on embedding-based link prediction. One line of such research focuses on the knowledge graphs, which can be viewed as a multi-relational graph composed of entities (nodes) and relations (different types of edges) [38,70,2,13,49]. Each edge is represented as a triple of the form (head entity, relation, tail entity). Link prediction in knowledge graph is typically referred to as the task of predicting an entity that has a specific relation with another given entity [66]. For example, (?, PresidentOf, USA) is to predict the president of USA. Another line centers on plain graphs, as the topic in this paper. The settings are also similar to what we have discussed in Section 3.4, i.e., to remove a small amount of the links and use the embeddings to predict the removed links [21,50]. The difference is that previous research overwhelmingly employ the Area Under the Curve (AUC) [16] as a metric for evaluation. AUC can be interpreted as the expectation that a target link is predicted with a higher probability than a randomly chosen non-existent link. However, considering the sparse feature of graphs, there are dramatically larger number of non-existent links than the number of removed links. Consequently, AUC is not an unbiased metric for evaluation and it is much easy to achieve a high score based on it [43].

To the best of our knowledge, there is few research for studying the divergence of the original link distribution and the embedding-derived distribution and the consistency of the communities discovered from the graph and embeddings.

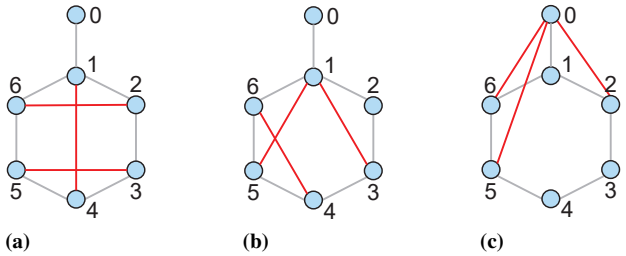


Fig. 3. Examples of graph reconstruction. (a) The original graph. (b) A reconstructed graph that has similar structure as the original one. (c) A reconstructed graph that has, to some extent, changed the structure of the original one. The red color emphasizes the difference between the three graphs.

6. Conclusion

We studied how well the graph topological structure is preserved by the embeddings from four aspects: 1) graph reconstruction based on the embeddings, 2) the divergence of the original link distribution and the embedding-derived distribution, 3) the consistency of the communities discovered from graph and embeddings, and 4) link prediction based on the embeddings. We did experiments on 12 graphs for 6 state-of-the-art graph embedding approaches. We found that the embeddings by these approaches can only preserve part of the topological structure. It is insufficient to rely on the embeddings to reconstruct the original graph, to discover communities, and to predict links at a high precision. This suggests that although the current graph embedding techniques can benefit graph analysis tasks such as label classification, we still cannot employ them for applications such as graph compression.

Graph embedding is not perfectly solved and there is still some room for improvement. Most of the embedding approaches ignore the hubness phenomenon that results in the heavy-tail degree distribution [54]. How to effectively utilize the dimensionality of the embeddings to encode the heavy-tail degree distribution will be left for our future work.

On the other hand, the proposed evaluation methods could be a standard for studying the problem of graph reconstruction or graph compression based on the embeddings, and be a benchmark for graph embedding approaches.

Acknowledgments. This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We would like to thank JSPS Grant-in-Aid for Early-Career Scientists (Grant Number 19K20352), JSPS Grant-in-Aid for Scientific Research(B) (Grant Number 17H01785) and JST CREST (Grant Number JPMJCR1687).

References

1. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. p. P10008 (2008)

2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Proceedingss of NIPS. pp. 2787–2795 (2013)
3. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: problems, techniques and applications. IEEE Transactions on Knowledge and Data Engineering 30(9), 1616–1637 (2018)
4. Cao, S., Lu, W., Xu, Q.: Grarep: Learning graph representations with global structural information. In: Proceedings of CIKM. pp. 891–900 (2015)
5. Cao, S., Lu, W., Xu, Q.: Deep neural networks for learning graph representations. In: Proceedings of AAAI. pp. 1145–1152 (2016)
6. Cavallari, S., Zheng, V.W., Cai, H., Chang, K.C., Cambria, E.: Learning community embedding with community detection and node embedding on graphs. In: Proceedings of CIKM. pp. 377–386 (2017)
7. Chang, S., Han, W., Tang, J., Qi, G.J., Aggarwal, C.C., Huang, T.S.: Heterogeneous network embedding via deep architectures. In: Proceedings of KDD. pp. 119–128 (2015)
8. Chen, S., Niu, S., Akoglu, L., Kovačević, J., Faloutsos, C.: Fast, warped graph embedding: Unifying framework and one-click algorithm. arXiv preprint arXiv:1702.05764 (2017)
9. Cheng, K., Li, J., Liu, H.: Unsupervised feature selection in signed social networks. In: Proceedings of KDD. pp. 777–786 (2017)
10. Choong, J.J., Liu, X., Murata, T.: Learning community structure with variational autoencoder. In: Proceedings of ICDM. pp. 69–78 (2018)
11. Dai, Q., Li, Q., Tang, J., Wang, D.: Adversarial network embedding. In: Proceedingss of AAAI. pp. 2167–2174 (2018)
12. Danon, L., Duch, J., D.-Guilera, A., Arenas, A.: Comparing community structure identification. J. Stat. Mech. p. P09008 (2005)
13. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2d knowledge graph embeddings. In: Proceedingss of AAAI. pp. 1811–1818 (2018)
14. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of KDD. pp. 135–144 (2017)
15. Feng, R., Yang, Y., Hu, W., Wu, F., Zhuang, Y.: Representation learning for scale-free networks. In: Proceedingss of AAAI. pp. 282–289 (2018)
16. Fielding, A.H., Bell, J.F.: A review of methods for the assessment of prediction errors in conservation presence/absence models. Environmental conservation 24(1), 38–49 (1997)
17. Fred, A.L.N., Jain, A.K.: Robust data clustering. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 128–133. Madison, WI, USA (Jun 2003)
18. Fu, T.y., Lee, W.C., Lei, Z.: Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In: Proceedings of CIKM. pp. 1797–1806 (2017)
19. Gao, M., Chen, L., He, X., Zhou, A.: Bine: bipartite network embedding. In: Proceedingss of SIGIR (2018)
20. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems 151, 78–94 (2018)
21. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of KDD. pp. 855–864 (2016)
22. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of NIPS. pp. 1025–1035 (2017)
23. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: methods and applications. IEEE Data Engineering Bulletin 40, 52–74 (2017)
24. Hu, J., Cheng, R., Huang, Z., Fang, Y., Luo, S.: On embedding uncertain graphs. In: Proceedings of CIKM. pp. 157–166 (2017)
25. Huang, X., Li, J., Hu, X.: Accelerated attributed network embedding. In: Proceedings of SDM. pp. 633–641 (2017)

26. Huang, X., Li, J., Hu, X.: Label informed attributed network embedding. In: Proceedings of WSDM. pp. 731–739 (2017)
27. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification 2(1), 193–218 (1985)
28. Jacob, Y., Denoyer, L., Gallinari, P.: Learning latent representations of nodes for classifying in heterogeneous social networks. In: Proceedings of WSDM. pp. 373–382 (2014)
29. Katz, L.: A new status index derived from sociometric analysis. Psychometrika 18(1), 39–43 (1953)
30. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: Proceedings of ICLR (2017)
31. Koutra, D., Shah, N., Vogelstein, J.T., Gallagher, B., Faloutsos, C.: Deltacon: principled massive-graph similarity function with attribution. ACM Trans. Knowl. Discov. Data 10(3), 28:1–28:43 (2016)
32. Koutra, D., Vogelstein, J.T., Faloutsos, C.: Deltacon: A principled massive-graph similarity function. In: Proceedings of SDM. pp. 162–170 (2013)
33. Kullback, S., Leibler, R.A.: On information and sufficiency. The Annals of Mathematical Statistics 22(1), 79–86 (1951)
34. Lai, Y.A., Hsu, C.C., Chen, W., Yeh, M.Y., Lin, S.D.: Preserving proximity and global ranking for node embedding. In: Proceedings of NIPS (2017)
35. Li, H., Wang, H., Yang, Z., Odagaki, M.: Variation autoencoder based network representation learning for classification. In: Proceedings of ACL Workshop. pp. 56–61 (2017)
36. Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., Liu, H.: Attributed network embedding for learning in a dynamic environment. In: Proceedings of CIKM. pp. 387–396 (2017)
37. Li, Y., Sha, C., Huang, X., Zhang, Y.: Community detection in attributed graphs: An embedding approach. In: Proceedings of AAAI. pp. 338–345 (2018)
38. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedingss of AAAI. pp. 2181–2187 (2015)
39. Liu, X., Kertkeidkachorn, N., Murata, T., Kim, K.S., Leblay, J., Lynden, S.: Network embedding based on a quasi-local similarity measure. In: Proceedings of PRICAI. pp. 429–440 (2018)
40. Liu, X., Murata, T., Kim, K.S.: Measuring graph reconstruction precisions—how well do embeddings preserve the graph proximity structure? In: Proceedings of WIMS. pp. 25:1–4 (2018)
41. Liu, X., Murata, T., Kim, K.S., Kotarasu, C., Zhuang, C.: A general view for network embedding as matrix factorization. In: Proceedings of WSDM. pp. 375–383 (2019)
42. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2), 129–137 (1982)
43. Lobo, J.M., Jiménez-Valverde, A., Real, R.: Auc: a misleading measure of the performance of predictive distribution models. Global Ecology and Biogeography 17(2), 145–151 (2008)
44. Ma, J., Cui, P., Zhu, W.: Depthlgp: learning embeddings of out-of-sample nodes in dynamic networks. In: Proceedings of AAAI. pp. 370–377 (2018)
45. Ma, Y., Ren, Z., Jiang, Z., Tang, J., Yin, D.: Multi-dimensional network embedding with hierarchical structure. In: Proceedingss of WSDM. pp. 387–395 (2018)
46. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
47. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of NIPS. pp. 3111–3119 (2013)
48. Newman, M.E.J.: Networks: An Introduction. Oxford University Press, New York (2010)
49. Nguyen, D.Q., Nguyen, T.D., Nguyen, D.Q., Phung, D.: A novel embedding model for knowledge base completion based on convolutional neural network. In: Proceedingss of NAACL-HLT. pp. 327–333 (2018)
50. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: Proceedings of KDD. pp. 1105–1114 (2016)
51. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of KDD. pp. 701–710 (2014)

52. Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., Han, J.: An attention-based collaboration framework for multi-view network representation learning. In: Proceedings of CIKM. pp. 1767–1776 (2017)
53. Radev, D.R., Qi, H., Wu, H., Fan, W.: Evaluating web-based question answering systems. In: Proceedingss of LREC. pp. 1153–1156 (2002)
54. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research 11, 2487–2531 (2010)
55. Ribeiro, L.F.R., Saverese, P.H.P., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: Proceedings of KDD. pp. 385–394 (2017)
56. Tang, J., Qu, M., Mei, Q.: Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of KDD. pp. 1165–1174 (2015)
57. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of WWW. pp. 1067–1077 (2015)
58. Tang, L., Liu, H.: Relational learning via latent social dimensions. In: Proceedings of KDD. pp. 817–826 (2009)
59. Tang, L., Liu, H.: Scalable learning of collective behavior based on sparse social dimensions. In: Proceedings of CIKM. pp. 1107–1116 (2009)
60. Tu, C., Liu, H., Liu, Z., Sun, M.: Cane: Context-aware network embedding for relation modeling. In: Proceedings of ACL. pp. 1722–1731 (2017)
61. Tu, C., Zhang, W., Liu, Z., Sun, M.: Max-margin deepwalk: discriminative learning of network representation. In: Proceedings of IJCAI. pp. 3889–3895 (2016)
62. Tu, K., Cui, P., Wang, X., Wang, F., Zhu, W.: Structural deep embedding for hyper-networks. In: Proceedingss of AAAI. pp. 426–433 (2018)
63. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of KDD. pp. 1225–1234 (2016)
64. Wang, H., Wang, J., Wang, J., ZHAO, M., Zhang, W., Zhang, F., Xing, X., Guo, M.: Graphgan: graph representation learning with generative adversarial nets. In: Proceedings of AAAI. pp. 2508–2515 (2018)
65. Wang, H., Zhang, F., Hou, M., Xie, X., Guo, M., Liu, Q.: Shine: Signed heterogeneous information network embedding for sentiment link prediction. In: Proceedings of WSDM. pp. 592–600 (2018)
66. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge graph embedding: a survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering 29(12), 2724–2743 (2017)
67. Wang, S., Aggarwal, C., Tang, J., Liu, H.: Attributed signed network embedding. In: Proceedings of CIKM. pp. 137–146 (2017)
68. Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., Yang, S.: Community preserving network embedding. In: Proceedings of AAAI. pp. 203–209 (2017)
69. Xu, L., Wei, X., Cao, J., Yu, P.S.: On exploring semantic meanings of links for embedding social networks. In: Proceedings of WWW. pp. 479–488 (2018)
70. Yang, B., Yih, W.t., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: Proceedingss of ICLR (2015)
71. Yang, C., Liu, Z., Zhao, D., Sun, M., Chang, E.: Network representation learning with rich text information. In: Proceedings of IJCAI. pp. 2111–2117 (2015)
72. Yang, C., Sun, M., Liu, Z., Tu, C.: Fast network embedding enhancement via high order proximity approximation. In: Proceedings of IJCAI. pp. 3894–3900 (2017)
73. Yang, D., Wang, S., Li, C., Zhang, X., Li, Z.: From properties to links: deep network embedding on incomplete graphs. In: Proceedings of CIKM. pp. 367–376 (2017)
74. Yang, Z., Cohen, W., Salakhudinov, R.: Revisiting semi-supervised learning with graph embeddings. In: Proceedings of ICML. pp. 40–48 (2016)
75. Zhang, D., Yin, J., Zhu, X., Zhang, C.: Network representation learning: a survey. IEEE Transactions on Big Data (2018)

76. Zhang, J., Xia, C., Zhang, C., Cui, L., Fu, Y., Yu, P.S.: Bl-mne: emerging heterogeneous social network embedding through broad learning with aligned autoencoder. In: Proceedingss of ICDM. pp. 605–614 (2017)
77. Zhang, Y., Xiong, Y., Kong, X., Zhu, Y.: Learning node embeddings in interaction graphs. In: Proceedings of CIKM. pp. 397–406 (2017)
78. Zhang, Y., Lyu, T., Zhang, Y.: Cosine: community-preserving social network embedding from information diffusion cascades. In: Proceedings of AAAI. pp. 2620–2627 (2018)
79. Zhang, Z., Cui, P., Pei, J., Wang, X., Zhu, W.: Timers: error-bounded svd restart on dynamic networks. In: Proceedings of AAAI. pp. 224–231 (2018)
80. Zhou, T., Lü, L., Zhang, Y.C.: Predicting missing links via local information. Eur. Phys. J. B 71(4), 623–630 (2009)

Xin Liu is a researcher of The National Institute of Advanced Industrial Science and Technology (AIST). He received a Doctor degree in Computer Science from Tokyo Institute of Technology in 2011. His main research interests are graph mining, data mining, machine learning, and neural networks.

Chenyi Zhuang joined Artificial Intelligence Research Center (AIRC), AIST as a researcher in April 2018. Prior to that, during October 2017 and March 2018, he was a post-doctor in Kyoto University. He received the BS degree in SE from Nanjing University in 2011, the MS degree and PhD degree in Informatics from Kyoto University in 2014 and 2017, respectively. In between, from 2015 to 2018, he was also serving as a young scientist in Japan Society for the Promotion of Science (JSPS). His current research primarily involves structured data mining, machine learning and urban computing.

Tsuyoshi Murata is an associate professor of the department of computer science in Tokyo Institute of Technology. He has been doing research on artificial intelligence, especially complex networks, machine learning and data mining. He served as one of the directors of The Japanese Society for Artificial Intelligence from 2013 to 2015.

Kyoung-Sook Kim is now the team leader of Data Platform Research Team at the Artificial Intelligence Research Center (AIRC) of AIST in Japan. She served as a researcher of National Institute of Information and Communications Technology (NICT) in Japan from 2007 to 2014. She received my B.S., M.S., and Ph.D. Degrees in Computer Science from Pusan National University in Korea in 1998, 2001, and 2007, respectively. She is also serving as a co-chair of OGC Moving Features Standard Working Group and an expert of ISO TC204 WG3. Her research interests are in Geo-enabled Computing Framework based on GIS, Location-based Services, Spatiotemporal databases, Big data analysis, Machine learning, etc.

Natthawut Kertkeidkachorn received a Ph.D. degree in Informatics from Sokendai (The Graduate University for Advanced Studies), Japan in 2017. He is currently a researcher at Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology and a visiting researcher at National Institute of Informatics, Japan. His research interests include the Semantic Web, machine learning and natural language processing.

Received: October 1, 2018; Accepted: June 3, 2019.

Computer Science and Information Systems 16(2):615–638 https://doi.org/10.2298/CSIS181010012

On Approximate *k*-Nearest Neighbor Searches Based on the Earth Mover's Distance for Efficient Content-Based Multimedia Information Retrieval*

Min-Hee Jang[1], Sang-Wook Kim[2], Woong-Kee Loh[3†], and Jung-Im Won[4†]

1 Mobile Communication Division, Samsung Electronics Co, Korea
zzmini@agape.hanyang.ac.kr
2 Department of Electronics and Computer Engineering Hanyang University, Korea
wook@hanyang.ac.kr
3† Department of Software, Gachon University, Korea
wkloh2@gachon.ac.kr
4† Smart Computing Lab. Hallym University, Korea
jiwon@hallym.ac.kr

Abstract. The Earth Mover's Distance (EMD) is one of the most-widely used distance functions to measure the similarity between two multimedia objects. While providing good search results, the EMD is too much time consuming to be used in large multimedia databases. To solve the problem, we propose an approximate k-nearest neighbor (k-NN) search method based on the EMD. In the proposed method, the overhead for both disk accesses and EMD computations is reduced significantly, thanks to the approximation. First, the proposed method builds an index using the M-tree, a distance-based multi-dimensional index structure, to reduce the disk access overhead. When building the index, we reduce the number of features in the multimedia objects through dimensionality-reduction. When performing the k-NN search on the M-tree, we find a small set of candidates from the disk using the index and then perform the post-processing on them. Second, the proposed method uses the approximate EMD for index retrieval and post-processing to reduce the computational overhead of the EMD. To compensate the errors due to the approximation, the method provides a way of accuracy improvement of the approximate EMD. We performed extensive experiments to show the efficiency of the proposed method. As a result, the method achieves significant improvement in performance with only small errors: the proposed method outperforms the previous method by up to 67.3% with only 3.5% error.

Keywords: Earth mover's distance, content-based information retrieval, k-nearest neighbor query.

---

*This is an extended version of a WIMS'18 conference paper.

† Co-Corresponding authors: Jung-Im Won (Hallym University), Email address: jiwon@hallym.ac.kr
Woong-Kee Loh (Gachon University), Email address: wkloh2@gachon.ac.kr

1. Introduction

Due to the recent advances in digital technologies, an enormous number of multimedia objects are now available over the Internet. In order to find the specific objects needed by the users from such a huge multimedia pool, an efficient search technique is highly essential [1], [2], [3], [4], [28]. Content-based information retrieval (CBIR) is the search technique based on the contents of multimedia objects. It finds the multimedia objects similar to the given query object based on features such as color distribution, shape, and texture [1], [5], [6], [26]. Since the CBIR is given with a multimedia object as a query example, it is more intuitive than non-CBIR such as keyword-based retrieval [1].

There are two types of the CBIR, namely the range query and the *k*-nearest neighbor (*k*-NN) query [7], [8], [9], [10], [27]. The range query finds a set of objects whose distances from the query object are less than or equal to the given threshold θ. It has the weakness that it is difficult to find a proper θ, thereby returning too big or too small search results according to θ. In the worst case, the range query may return either an empty or the whole dataset [9], [11], [12]. The *k*-NN query finds *k* objects that are nearer from a query object than other objects. Since the *k*-NN query does not require a hard-to-estimate distance θ from the query, users can converge on their wanting objects more quickly. In this paper, we focus our attention on the *k*-NN CBIR query.

For the CBIR, a multimedia object can be represented as an *n*-dimensional probabilistic histogram [8], [12], [13]. An *n*-dimensional histogram *H* is composed of *n* bins, where each bin represents a probability p_i, i.e., $H = \{p_1, p_2, ..., p_n\}$. The probabilities p_i can be located in a multi-dimensional space rather than serially aligned. For example, an image with 10 colors in the *RGB* space is represented as a 10-dimensional histogram, i.e., $n = 10$. The weight of each color in the image is represented as the probability of the corresponding bin in the histogram, and the red (*R*), green (*G*), and blue (*B*) components of each color are located in the three-dimensional space.

For the CBIR on multimedia databases, we need a distance function which measures the similarity between two objects. There have been a number of distance functions proposed such as the Euclidean distance or the $\chi 2$ statistic. While they can be computed very fast and give good results in some cases, they do not take into account all possible variations of matching, which could cause inaccurate search results [14]. The Earth mover's distance (EMD) is a representative distance function for the CBIR [8], [11], [12], [13] to address this problem [14], [15]. The EMD between two histograms is defined as the minimum work needed to transform one histogram into the other by moving portions of bins [14]. The work is defined as the weight (portion) of a bin moved times the moving distance. Due to the high quality of the search results based on the EMD, it is adopted in various applications [14], [15]. However, the time complexity of EMD computation is $O(n^3 \log n)$, where *n* is the number of bins in a histogram, and thus the EMD cause significant overhead for large multimedia databases [8], [12], [14].

Many efforts have been made for efficient EMD-based CBIR: approximation methods [15], [16], lower-bound methods [17], [18], and indexing methods [8], [12]. The approximation methods compute the approximate EMD very quickly. They improved the EMD computation time in an order of magnitude; however, they have scalability problems that incur large disk access overhead. The lower-bound methods first find a set of candidates using lower-bound functions and then compute the actual EMD between the candidates and the query object to find the real *k*-NN results. They significantly reduce the number of the EMD computations owing to the lower-bound

functions; but they also have scalability problems. The indexing methods are proposed to reduce both the disk access overhead and the computational overhead. They first find a set of candidates using the indexes and then compute the actual EMD between the candidates and the query to return the final *k*-NN result. By using the indexes, they can reduce the disk access overhead. However, they still require a considerable number of costly EMD computations in the post-processing step, because the number of candidates is much larger than *k*. Therefore, for efficient EMD-based CBIR, both the disk access overhead and the EMD computation overhead should be reduced at the same time.

In this paper, we propose an efficient approximate *k*-NN method based on the EMD. The proposed method uses the M-tree [7], a distance-based multi-dimensional index, to reduce the disk access overhead. The M-tree is constructed using distances between objects rather than the actual object positions in the multi-dimensional space. The M-tree generally provides good search performance for retrieving high-dimensional histograms and thus reduces the disk access overhead of the EMD-based CBIR [7], [17]. When the CBIR is done using the M-tree, it requires a large number of EMD computations between the query object and those in the index [7], [17]. Since the complexity of EMD computation is very high, it should incur a serious computational overhead and thus dramatic deterioration of the search performance. To solve the problem, we reduce the dimensionality of *n*-dimensional histogram into n' ($<< n$) through dimensionality-reduction [18] and the M-tree is constructed using the dimension-reduced histograms.

Besides, we use the approximate EMD [16] to reduce the EMD computation overhead when retrieving the M-tree. The approximation EMD, which is very close to real EMD, is computed in a linear time with a small error. Since the approximate EMD between two dimension-reduced histograms is always lower than the approximate EMD between the original histograms, there occurs no false-dismissal, which will be proven in Section 4.2. Since the M-tree is constructed using the dimension-reduced histograms, a set of candidates are obtained as the result of the M-tree search. Then, the post-processing is performed to find the real *k*-NN result by computing the EMD on original histograms of the candidates. Even in the post-processing step, a large number of EMD computations should be performed because the number of candidates is much larger than *k*. The proposed method uses the approximate EMD to speed up the post-processing. To minimize the errors due to the approximate EMD, we propose an accuracy improvement method. Extensive experiments on real datasets comparing our method with the state-of-the-art methods [8], [12], [18] were conducted to show the superiority of our method. The result reveals that our method outperforms the previous ones by up to 67.3% with the error as small as 3.5%.

This paper is organized as follows. Section 2 describes the EMD in more detail, and Section 3 briefly reviews the related work on the EMD-based CBIR and points out their weaknesses. Section 4 presents our method in detail and Section 5 evaluates its performance in comparison with existing ones. Finally, Section 6 summarizes and concludes this paper.

2. Earth Mover's Distance

Let $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$ ($\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i$) be two n-dimensional histograms, where p_i and q_i are bins in the histograms. A matrix $D = [d_{ij}]$ is called a ground distance matrix, where d_{ij} is the ground distance between p_i and q_j. The ground distance is defined by any standard metric, such as the Euclidean distance or the Manhattan distance [14]. A matrix $F = [f_{ij}]$ is called a flow matrix, where f_{ij} is the portion of bin (called weight or probability) transferred from p_i to q_j. The work is defined as the multiplication of flow f_{ij} and ground distance d_{ij}. The EMD between P and Q is defined as the minimum amount of work required to transform a distribution P into the other Q (or in the opposite direction) and formally written as:

$$EMD(P,\ Q) = min\left\{\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} \cdot d_{ij}\right\} \quad (\mathbf{1})$$

subject to:

$$\forall i, j : f_{ij} \geq 0,$$

$$\forall i : \sum_{j=1}^{n} f_{ij} = p_i,\ and$$

$$\forall j : \sum_{i=1}^{n} f_{ij} = q_j$$

Figure 1 shows an example of computing the EMD between P and Q, which are represented as 6-dimensional histograms (i.e., $n = 6$). In the figure, the x-axis represents each bin in the histograms and the y-axis represents the weight (or probability) of each bin. When transforming P into Q, the portions of bins in P can be moved to different positions to match the bins in Q in a variety of ways. In Figure 1, the amount of work is the minimum when each bin portion in P is moved to the position designated with the same character in Q. Thus, the EMD is computed as:

$$\text{EMD}(P,Q) = 0.3 \times 2\big(Work(A)\big) + 0.3 \times 1(Work(B)) + 0.2 \times 3(Work(C)) + 0.2 \times 1(Work(D))$$

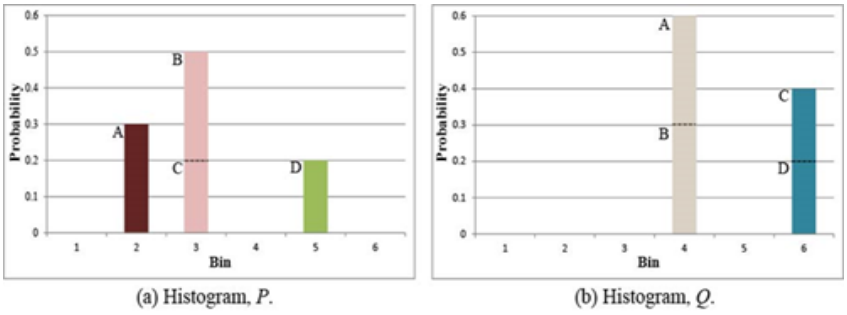


Fig. 1. Example of computing the EMD between two 6-dimensional histograms.

3. Related Work

3.1. Approximate EMD

The approximate EMD (AEMD) between two multimedia objects is computed in a time linear to the number of bins, and it is very close to the actual EMD. In a one-dimensional space, the exact EMD between two histograms is computed as follows: (1) it selects the bins at the end in the one-dimensional space from each histogram and computes the work for the common weights of the selected bins; (2) the common weights are removed from the histogram; (3) the steps (1) and (2) are repeated until all bins in two histograms are removed. While repeating the steps (1)~(3), the (partial) works are all added, and the final sum is returned as the EMD between two histograms. Since both bins are scanned only once, the time complexity of one-dimensional EMD computation is O(n) [16], which is much smaller than that of multi-dimensional EMD computation. Based on this idea, AEMD linearizes the multi-dimensional histogram using the Hilbert curve. The Hilbert curve fills the multi-dimensional space with a continuous curve as shown in figure 2 [19]. After the linearization, AEMD applies the method mentioned above.

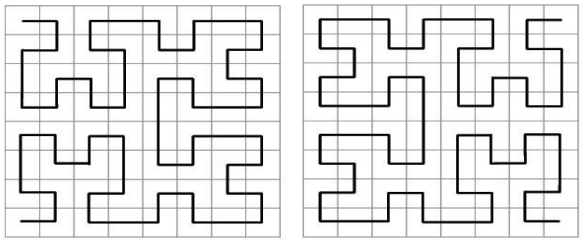


Fig. 2. Example of the Hilbert curve in two-dimensional space.

To pursue the highest accuracy, AEMD creates multiple Hilbert curves with different starting points and directions. In an m-dimensional space, there can be $(2^m \cdot m!)/2$ Hilbert curves [16]. AEMD takes the minimum of the works obtained for each of the curves. As an experimental result in [16], AEMD incurred an error up to only 4.2%, while it dramatically improved the speed by up to 34 times over the EMD. However, AEMD suffers from the scalability problem, since it should access all the objects in a multimedia database for CBIR.

3.2. Lower-bound Based on Dimensionality Reduction

As indicated in the complexity, the time for EMD computation for lower dimensional histograms should be smaller than that for higher dimensional ones. The method by Wichterich, et al. (called *LB* in this paper) transforms an n-dimensional histogram H

into an n'-dimensional one $H'(n' < n)$ using an $n \times n'$ reduction matrix $R = [r_{ij}]$ as the following equation [18]:

$$H' = H \cdot R \quad (2)$$

where it holds that:

$$\forall i,j \left(1 \leq i \leq n,\ 1 \leq j \leq n'\right),\ r_{ij} \in \{0,\ 1\},$$

$$\forall i,\ \sum_{j}^{n'} r_{ij} = 1,\ and$$

$$\forall j,\ \sum_{i}^{n} r_{ij} \geq 1$$

LB computes the ground distance between the bins in the dimensionality-reduced histograms using the *optimal reduced distance matrix* $D' = [d'_{ij'}]$. An element $d'_{ij'}$ in the matrix is defined as:

$$d'_{i'j'} = min\{d_{ij} | r_{ii'} = 1 \wedge r_{jj'} = 1\} \quad (3)$$

The optimal reduced distance matrix D' ensures that the EMD between the dimensionality-reduced histograms is always lower than (or lower-bounds) that between the original histograms [18]. The main advantages of this dimensionality reduction include efficiency, flexibility, and completeness [18]. By using this lower-bound property, LB scans all the objects in the database and returns a set of candidates. Then, LB computes the EMD between the candidate's original histogram H and the query histogram, and returns the qualified ones. This method reduces EMD's computational overhead; however, as the AEMD, it also suffers from the scalability problem, since it should access all the objects in the database.

3.3. Indexing Methods

To solve the scalability problem, a few indexing methods such as the tree-based index (called *TBI* in this paper) [8] and the normal lower-bound index (called *NLI* in this paper) [12] have been proposed. TBI employs LB^+-trees; every histogram is mapped to L domain values using the primal-dual theorem [20], and then each set of l-th ($0 \leq l < L$) values is indexed in a B^+-tree. TBI processes range queries and k-nearest queries efficiently using the trees. TBI converts the EMD-based range query into L range queries against L trees, and the objects contained in the intersection of the results obtained through the trees constitute the candidate set. Then, TBI computes the actual EMD for each candidate object to find the final query result. In TBI, the number of B^+-trees increases as the database size increases. To solve this problem, NLI employs a single Quad-tree to index multi-dimensional histograms [12]. NLI projects a histogram onto a vector and approximates it by a normal distribution. NLI then represents each normal as a point in a Hough transformed space and stores in the Quad-tree. NLI computes the lower-bound EMD in a constant time. Since NLI is based on

approximation, it still needs EMD computations in the post-processing phase on the candidates obtained through the index.

Although these methods reduce the number of EMD computations, they still require a considerable number of EMD computations in the post-processing phase. As a result of our experiment on a database composed of 3,932 high-dimensional histograms [8], [12], TBI and NLI should have performed about 350 and 300 EMD computations, respectively. The EMD computations significantly deteriorate the performance of the EMD-based CBIR. Recently, a refinement method [21] has been proposed to reduce the post-processing overhead by adopting the simplified graph incremental algorithm and the dynamic refinement order. However, the method focuses only on the post-processing and can be applied only to NLI, unlike our method explained in the next section.

4. Proposed Method

In this section, we propose an efficient approximate k-NN algorithm for EMD-based CBIR. In Section 4.1, we present a naïve approach incurring no false dismissal. In Sections 4.2 and 4.3, we refine the naïve approach for performance improvement and accuracy improvement.

4.1. Naïve Approach

The naïve approach performs the *k*-NN using the index for scalability. Since we deal with the *n*-dimensional histograms, we may consider the R-tree [22], a most widely used multi-dimensional index. However, the R-tree suffers from the so called high-dimensionality problem or high-dimensionality curse [7]: the performance of the R-tree rapidly deteriorates with the increase of the histogram dimension. If the R-tree is employed for the EMD-based CBIR, the search performance using the R-tree becomes worse than even the sequential scan when the dimension of the histograms exceeds 35 [13].

In this paper, we adopt the M-tree, a distance-based index structure [7]. While the R-tree is constructed based on the object locations in the multi-dimensional space, the M-tree is based on the distances between the objects. The distance computations for only a small number of (not all possible) object pairs are required to construct the M-tree [7]. Also, the M-tree provides good performance not only in the indexing but also in the search for high-dimensional histograms, because the M-tree performs the search using the distances previously computed when constructing the index. Figure 3 shows an example of the M-tree. Figure 3(a) shows the distribution of objects ($o_1 \sim o_{10}$) in the two-dimensional space, and Figure 3(b) shows an M-tree constructed based on distances between these objects in Figure 3(a). In the figure, we set the maximum number of entries of the internal and terminal nodes are two and three, respectively.

The naïve approach constructs the M-tree index based on the EMD between *n*-dimensional histograms and performs the *k*-NN search using the *k*-NN algorithm for the M-tree presented in [7]. The naïve approach incurs no false dismissal, since EMD satisfies the following three conditions:

(1) Positivity: $\mathrm{EMD}(P, Q) \geq 0$,
(2) Symmetry: $\mathrm{EMD}(P, Q) = EMD(Q, P)$, and
(3) Triangular inequality: $\mathrm{EMD}(P, Q) \leq \mathrm{EMD}(P, R) + \mathrm{EMD}(R, Q)$

where P, Q, and R are histograms.

However, the naïve approach has the performance problem because the EMD has the very high computation complexity $O(n^3 \log n)$ for n-dimensional histograms [14]. Since many EMD computations are required in the course of the indexing and the searching, it causes serious deterioration of overall performance of the approach. To solve this problem, we provide three extensions as follows. First, we use the dimensionality-reduction [18] explained in Section 3.2. That is, we transform n-dimensional histograms to n'-dimensional ones ($n' << n$) using the dimensionality-reduction when constructing the M-tree and performing the k-NN search. Since the dimension of histograms is reduced significantly, the EMD computation overhead should also be reduced as much. Second, we use the AEMD explained in Section 3.1 instead of the EMD [16]. The AEMD is computed in $O(n)$ time with small errors with the EMD. Third, we adopt the maximum common weight elimination to reduce the error between the AEMD and the EMD. We explain first and second extensions in Section 4.2 and third one in Section 4

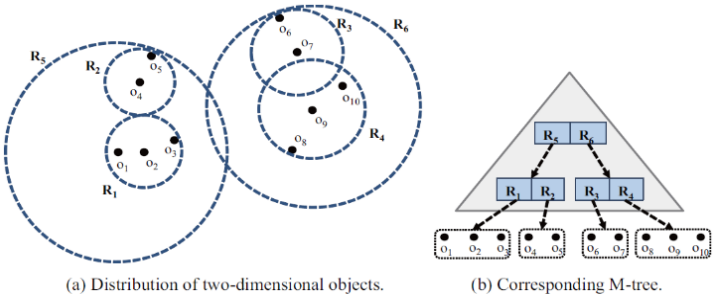


Fig. 3. A Sample M-tree.

4.2. Performance Improvement

The proposed k-NN method adopts the dimensionality-reduction [18] using the reduction matrix R for both the indexing and the searching. When indexing, we compute the EMD with the optimal reduced distance matrix D' to construct the M-tree. Likewise, when searching, we reduce the dimension of the query histogram by using same matrix R and perform the k-NN search on the M-tree with D' and the EMD. Since $n' << n$, this method performs the indexing and searching more efficiently than the naïve approach. Note that the method based on the dimensionality-reduction incurs no false dismissal. For its justification, we need the following lemma:

Lemma 1. For any two n-dimensional histograms P and Q, the following is satisfied [18]:

$$EMD_{D'}(P', Q') \leq EMD_D(P, Q) \quad \textbf{(4)}$$

where $EMD_{D'}(P', Q')$ is the EMD with the optimal reduced distance matrix D' after reducing the dimension of P and Q into P' and Q' using the reduction matrix R, and $EMD_D(P, Q)$ is the EMD with the original ground distance matrix D and the original histograms P and Q.

Proof: See [18]. □

As shown in Eq. (4), the EMD between any two dimensionality-reduced histograms lower-bounds the EMD between original histograms. Based on this lower-bounding property, we introduce a *k*-NN algorithm as shown in Figure 4.

The figure shows the *k*-NN_EMD algorithm based on the dimensionality-reduction, and the M-tree that is constructed with the dimensionality-reduced histograms in the algorithm. The *k*-NN_EMD is given a query histogram Q and the number of objects k as the input and returns an array of the *k*-NN objects as the output. In line (1), the algorithm calls *k*-NN_Search() function for the M-tree. This function is presented in [7] and performs *k*-NN search using the M-tree. Since the M-tree is constructed with the dimensionality-reduced histograms, we also reduce the dimension of the query Q into Q' and find the *k*-NN objects by invoking the *k*-NN_Search(). In line (2), we compute the original EMD, $EMD_D(Q, P)$ for each P returned from the *k*-NN_Search() and then, in line (3), we sort the result and save in the array NN. As a point, $NN[k]$ is the objects with the largest EMD from Q among all the objects in NN.

Then, we call Next-NN_Search() function with Q as the input. This function returns the nearest neighbor N next to the current nearest neighbors NN using the M-tree. For example, when we have *k*-NN objects obtained by searching the M-tree, if we call the Next-NN_Search() function, it returns the $(k+1)$-th object. If we call the Next-NN_Search() function again, it returns the $(k+2)$-th object. This function can be easily implemented using the *k*-NN_Search() function as follows. Let $k(0)$ be the predefined number of objects currently searched by the *k*-NN_Search(). When the Next-NN_Search() is called, it returns the i-th ($i \leq k(0)$) nearest neighbor among the $k(0)$ objects. If $i > k(0)$, the *k*-NN_Search() is invoked internally to search $k(1)$ ($> k(0)$) objects next farther than the $k(0)$ objects and the i-th ($i \leq k(1)$) object is returned. The Next-NN_Search() function returns the next nearest neighbor continuously by repeating this process.

In line (6), we compute $EMD_D(Q, N)$ of N and, if the result is smaller than the EMD of $NN[k]$, N is inserted into the NN preserving the order of EMD from the original Q, and the previous $NN[k]$ is discarded. We call Next-NN_Search() again in line (9). As shown in line (5), this process is repeated until $EMD_D(Q, N)$ is smaller than $EMD_{D'}(Q', NN[k]')$. If $EMD_D(Q, N)$ is larger than $EMD_{D'}(Q', NN[k]')$, we break the while loop and finally return the array NN to the user in line (11). In summary, the *k*-NN_EMD algorithm first searches for the *k*-NN candidates by using the M-tree constructed with the dimensionality-reduced histograms and then refines the *k*-NN result by comparing the actual EMD of the original high-dimensional histograms.

```
Algorithm k-NN_EMD(Q, k, NN)
Q: query histogram (Input)
k: number of neighbors (Input)
NN: array of k-nearest neighbors (Output)
1:    Call k-NN_Search(Q', k);
2:    Compute EMD_D(Q, P) of each object in k-NN
3:    Save the output to NN in the order of EMD from original Q;
4:    Call Next-NN_Search(Q) and let N be the result;
5:    while EMD_D(Q, N) ≤ EMD_D(Q', NN[k]')
6:        if EMD_D(Q, N) ≤ EMD_D(Q, NN[k])
7:            Insert N into NN preserving the order of EMD from original Q;
8:        end if
9:        Call Next-NN_Search(Q) and let N be the result;
10:   end while
11:   return NN;
```

Fig. 4. k-NN_EMD algorithm based on dimensionality reduction.

The *k*-NN_EMD guarantees no false dismissal, and it can be easily shown using Eq. (4) as follows. *NN*[*k*] is the object with the farthest EMD from *Q* in *NN* returned from the *k*-NN_EMD (in line (11)). For the next nearest neighbor *N* given by the Next-NN_Search(), it holds that EMD_D (*Q*, *NN*[*k*]) ≤ $EMD_{D'}$ (*Q'*, *N'*) according to the condition in line (5). It also holds that $EMD_{D'}$ (*Q'*, *N'*) ≤ EMD_D (*Q*, *N*) according to Eq. (4). Therefore, for any *N* in the M-tree, it holds that EMD_D (*Q*, *P*) ≤ EMD_D (*Q*, *N*) indicating that the original EMD of any object in the M-tree is larger than the original EMD of any *P* in *NN*, and thus *NN* is the set of *kNN* from *Q*.

In the *k*-NN_EMD, we use the dimensionality-reduced histograms for efficient indexing and searching; however, there is still performance deterioration even with the dimensionality-reduced histograms due to very high complexity of the EMD. Moreover, many EMD should also be computed for original histograms when refining the candidates. To solve this problem, instead of the EMD, we use the AEMD proposed by Jang et al. [16] to improve the indexing and searching efficiency. According to the experiment results in [16], the AEMD computation requires only O(*n*) time with only 4.2% average error, where the average error is defined as $\left|\frac{EMD(P,Q)-AEMD(P,Q)}{EMD(P,Q)}\right|$.

By adopting the AEMD, the proposed method performs the approximate *k*-NN search. Figure 5 shows the *k*-NN_AEMD algorithm which is a modification of the *k*-NN_EMD algorithm in Figure 4. In this algorithm, we assume the size of NN to be *k'* (> *k*).

As in the *k*-NN_EMD, the *k*-NN_AEMD is given a query histogram *Q* and the number of objects *k* as the input and returns an array *NN* of *k*-NN objects based on the EMD. The main difference of the *k*-NN_AEMD from the *k*-NN_EMD is that the *k*-NN_AEMD constructs the M-tree using the EMD but searches the tree using the AEMD. It is for reducing both the accuracy loss due to the AEMD and the computational overhead due to the EMD. If we use only the EMD as shown in Figure 4, it returns the exact *k*-NN result based on the EMD with poor search performance.

```
Algorithm k-NN_AEMD(Q, k, NN)
Q: query histogram (Input)
k: number of neighbors (Input)
NN: array of k-nearest neighbors (Output)
 1:   Compute k' as a function of k;
 2:   Call k-NN_Search(Q', k');
 3:   Compute AEMD_D (Q, P) of each object in k'-NN
 4:   Save the output to NN in the order of AEMD from original Q;
 5:   Call Next-NN_Search(Q) and let N be the result;
 6:   while AEMD_D (Q, N) ≤ AEMD_D(Q', NN[k'])
 7:        if AEMD_D(Q, N) ≤ AEMD_D(Q, NN[k'])
 8:             Insert N into NN preserving the order of AEMD from original Q;
 9:             Call Next-NN_Search(Q) and let N be the result;
10:        end if
11:   end while
12:   Post-process NN;
13:   return NN;
```

Fig. 5. k-NN_AEMD algorithm using AEMD.

On the contrary, if we use only the AEMD for both the indexing and the searching, it returns highly inaccurate results with little performance improvement over the *k*-NN_AEMD for the following reason. The search on the M-tree is based on the distances previously computed when constructing the M-tree. If we construct the M-tree using the AEMD, the distances have errors. When the M-tree is used to compute the distances between the objects in the course of *k*-NN search, the distances based on the AEMD causes additional errors. Thus, the errors are accumulated, which causes a lot of unexpected false dismissals. Therefore, we decide to use the AEMD only in the search process to minimize false dismissals.

To minimize the false dismissals due to the AEMD, we perform the *k'*-NN search instead of the *k*-NN ($k' > k$) in line (1). The *k*-NN_AEMD algorithm after the line (1) is almost the same as the *k*-NN_EMD algorithm in Figure 4 and performs the *k'*-NN search down to line (11). In line (12), the post-processing is performed to return the *k*-NN result to the user: the original EMD is computed for each object in the *k'*-NN array from the query *Q*, and *k* objects with the smallest EMD are returned in line (13). This method reduces the EMD computation overhead, because it only needs to compute the EMD for *k'* candidates in the post-processing step. There is additional computation overhead when searching *k'*-NN objects from the M-tree for *k'* larger than *k*; however, it is only fairly marginal compared with the overall performance gain obtained by the AEMD. We discuss in detail on the accuracy and the performance of the *k*-NN_AEMD algorithm in Section 5. As in the *k*-NN_EMD, the *k*-NN_AEMD incurs no false dismissal by the dimensionality-reduction. That is, the *k'*-NN result found in the *k*-NN_AEMD until the line (11) contains every object that should be eventually returned as the search result. For the justification, we need the following lemma:

Lemma 2. For any two *n*-dimensional histograms *P* and *Q*, the following is satisfied:

$$AEMD_{D'}(P', Q') \leq AEMD_D(P, Q) \quad (5)$$

where $AEMD_{D'}(P', Q')$ is the AEMD with the optimal reduced distance matrix D' after reducing the dimension of P and Q into P' and Q' using the reduction matrix R, and $AEMD_D(P, Q)$ is the AEMD with the original ground distance matrix D and the original histograms P and Q.

Proof: See the Appendix. □

We can show using Lemma 2 that no false dismissal is caused by the dimensionality-reduction in the k-NN_AEMD algorithm. This proof is almost the same as in the k-NN_EMD. $NN[k']$ is the farthest object from Q in NN obtained until the line (11) in Figure 5. For the next nearest neighbor N given by the Next-NN_Search(), it holds that $AEMD_D(Q, NN[k]) \leq AEMD_{D'}(Q', N')$ according to the condition in line (6). It also holds that $AEMD_{D'}(Q', N') \leq AEMD_D(Q, N)$ according to Eq. (5). Thus, for any object N in the M-tree, it holds that $AEMD_D(Q, P) \leq AEMD_D(Q, N)$. It indicates that the original AEMD of any object in the M-tree is larger than the original AEMD of any P in NN, and thus NN is the set of k'-NN from Q.

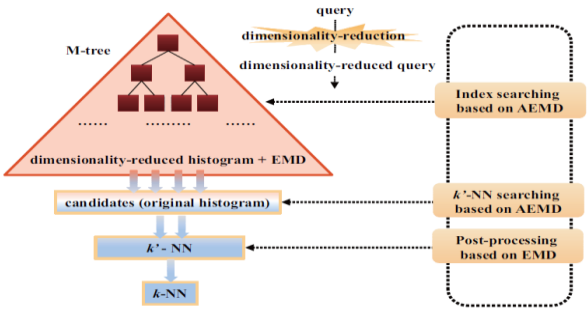


Fig. 6. Indexing and searching in the k-NN_AEMD.

Figure 6 shows the process of the k-NN_AEMD algorithm. In the figure, the M-tree is constructed based on the EMD of the dimensionality-reduced histograms. When a query Q is issued, the algorithm first reduces the dimension of the query histogram in the same manner. The algorithm finds the k'-NN objects based on the AEMD of the dimensionality-reduced histograms. Then, it finds the set of k'-NN candidates by searching the M-tree based on the AEMD of the original histograms. Finally, the original EMD for each k'-NN candidate is computed in the post-processing step, and the k-NN objects with the smallest EMD from Q are returned to the user.

4.3. Accuracy Improvement

The *k*-NN_AEMD finds k' (> k) nearest neighbors to reduce the false dismissals due to the error between the AEMD and the EMD. However, even the k'-NN search with very large k' might cause false dismissals for very high dimension histograms because the error between the AEMD and the EMD increases as the dimension increases. To reduce such error due to the AEMD, we propose the elimination of maximum common weight (MCW). The MCW is the smaller weight of the two corresponding bins with the same location *i*. For two *n*-dimensional histograms $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, \ldots, q_n\}$, the MCW c_i ($1 \le i \le n$) is defined as $c_i = \min\{p_i, q_i\}$. The MCW-eliminated histograms P_c and Q_c are defined as follows:

$$P_c = \{p_1 - c_1, \cdots, p_n - c_n\} \text{ and } Q_c = \{q_1 - c_1, \cdots, q_n - c_n\}$$

When computing the EMD, the work of the MCW of two corresponding bins is 0, because the ground distance of the bins with the same location is 0. Since the EMD is defined as the minimum work, we get the same EMD, whether the MCW elimination is applied or not. On the contrary, when computing the AEMD, since the MCW bins might move to different locations, the work of the MCW of the two corresponding bins could be larger than 0. To solve this problem, we eliminate the MCW before the AEMD computation. That helps reduce the error between the AEMD and the EMD significantly. Our experiment result shows that the average error of the AEMD is reduced to half by applying the MCW elimination. We discuss this result in detail in Section 5.

The MCW elimination is applied in all the lines that compute the AEMD, i.e., in the lines (2) ~ (9) in the *k*-NN_AEMD algorithm in Figure 5. The lines (2), (5), (6) and (9) compute the AEMD for the n'-dimensional histograms, and the lines (3), (4), (6), (7) and (9) for the *n*-dimensional histograms. The overhead of the MCW elimination is trivial since it only needs simple arithmetic operations in the O(n) time.

Another problem that causes the error between the AEMD and the EMD is that the AEMD does not generally satisfy the triangular inequality explained in Section 4.1, while the positivity and the symmetry are satisfied. The false dismissals can be generated due to the problem in the *k*-NN_AEMD algorithm. However, our experiment result with all possible histogram pairs in a real dataset shows that only 0.2% of the pairs violate the triangular inequality. Therefore, we can claim that the false dismissals due to the violation of triangular inequality are not significant. The detail is discussed in Section 5.2.

5. Evaluation

5.1. Experimental Setup

In this section, we verify the superiority of the proposed method through extensive experiments. We used three datasets in the experiments, namely RETINA [8], [12],

[18], Shader [23], and SIMPLIcity [16], [24]. RETINA is a high-resolution image set which consists of 3,932 feline retina scans. Each image is represented as a 96-dimensional histogram, and each bin in the histogram represents a location in two-dimensional space. Shader contains 30,000 computer graphics (CG) images. Each image is represented as a 128-dimensional histogram with the bins in three-dimensional space. For each image, the weight of each of 128 RGB colors was extracted and used as histogram bins. SIMPLIcity contains 1,000 images in 10 categories, and each category has 100 images. The images in this dataset are also represented as 128-bin histograms with the bins in three-dimensional space. We compared the performance of our method with three previous methods: lower-bounding based on dimensionality reduction (LB) [18], tree-based indexing (TBI) [8], and normal lower-bound indexing (NLI) [12]. For dimensionality-reduction in LB, we set the reduced histogram dimension n' that showed the best search performance in the experiments in [18]. We set n'= 18 for RETINA and n'=24 for Shader and SIMPLIcity, respectively. The same n' values were used in the proposed method. Table 1 shows the size of the M-tree indexes constructed in the proposed method.

Table 1. Size of the M-tree indexes for three datasets

	RETINA	SIMPLIcity	Shader
Data Size	1,391KB	412KB	12,154KB
Index Size	3,456KB	1,187KB	36,032KB

We used the precision, recall, and mean absolute percentage error (MAPE) as the accuracy measures. The precision and recall measures were used on the SIMPLIcity. As described above, the SIMPLIcity is composed of 10 categories, and each category contains 100 images. The category information is used as the ground truth to measure the precision and recall defined as follows [16]:

$$Precision = \left|\frac{\{data\ in\ each\ category\} \cap \{retrieved\ data\}}{\{retrieved\ data\}}\right|$$
$$,\quad Recall = \left|\frac{\{data\ in\ each\ category\} \cap \{retrieved\ data\}}{\{data\ in\ each\ category\}}\right| \tag{6}$$

The MAPE is used to evaluate the accuracy of the approximate method and defined as the following [16], [25]:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{a_i - f_i}{a_i}\right| \tag{7}$$

where a_i is the real EMD and f_i is the AEMD between two objects. In our experiments, the MAPE values are multiplied by 100 to show in the unit of percent. We used the total elapsed time as the performance measure. To obtain the more accurate results, we averaged the results on 100 random queries for each experiment. The experiments were performed on Microsoft Windows 7 on a workstation equipped with an Intel Core i5 2.80 GHz CPU and 4GB main memory.

5.2. Experimental Results

We evaluate the accuracy of the AEMD in the first and second experiments and then verify the superiority of the proposed method in the remaining experiments. We use only RETINA and Shader in evaluating the search performance, because SIMPLIcity has the histogram distribution similar to Shader and contains much smaller number of objects. As mentioned in Section 4.1, SIMPLIcity is used to measure the precision and recall of the proposed method. In the first experiment, we measured the MAPE of the AEMD, and the result is shown in Table 2.

Table 2. MAPE of AEMD

	RETINA	SIMPLIcity	Shader
AEMD	27.6%	21.4%	19.2%
AEMD (with MCW elimination)	14.4%	11.7%	9.8%

As shown in the table, the original AEMD incurs considerable errors for every dataset. Furthermore, the errors of the AEMD are larger than those reported in [16] because our datasets consist of higher dimensional histograms. In contrast, the errors of the AEMD are reduced nearly to half when the MCW elimination was employed. This is because the MCW elimination removes the histogram bins that may cause the errors in the AEMD computations. In all the experiments hereafter, we used the AEMD with the MCW elimination.

In the second experiment, we measured the ratio of object pairs satisfying the triangular inequality of the AEMD. As mention in Section 4.2, the AEMD does not generally satisfy the triangular inequality due to the errors between the AEMD and the EMD. We used Shader since its size is the largest. The experiment was performed with respect to a varying number of objects pairs, and its result is shown in Table 3.

Table 3. MAPE of AEMD

Number of comparisons	1,000	5,000	10,000	20,000	30,000
Satisfaction ratio	100%	99.9%	99.9%	99.8%	99.8%

As shown in Table 3, there exist only few cases violating the triangular inequality. This is because the error between the EMD and the AEMD for a pair (P, R) is not larger than the errors generated by the pairs (P, Q) plus (Q, R). In order to examine in a different viewpoint, we calculated the standard deviation of the MAPE, and the result is shown in Table 4. We can find in the table that the standard deviation of the errors is quite small. That is, the errors by the AEMD are mostly constant. Therefore, the false dismissals by the AEMD by violating the triangular inequality are not significant.

Table 4. Standard deviation of MAPE

	RETINA	SIMPLIcity	Shader
AEMD	0.0343	0.0320	0.0303
AEMD(with MCW elimination)	0.0235	0.0216	0.0211

In the third experiment, we measured the search performance and the accuracy of the proposed method while changing the parameter k'. The proposed method searches k' ($> k$) candidates based on the AEMD in the M-tree and then performs the post-processing based on the EMD to find k objects nearest from the query. In this experiment, we set k as 20 and the accuracy is defined as the ratio of common objects between the k-NN result based on the EMD and the search result of the proposed method. Table 5 shows the result. The result shows that both the accuracy and the elapsed time increase as k' increases. The accuracy is improved with the increase of k' because the false dismissals due to the AEMD are reduced. The execution time also increases slowly with the increase of k' due to the fast computation of the AEMD. In the experiments hereafter, we set $k' = 4k$ for RETINA and $k' = 2k$ for Shader.

Table 5. Accuracy and search performance of the proposed method with respect to varying *k'* values

(a) RETINA

	k'=k	*k'=2k*	*k'=3k*	*k'=4k*	*k'=5k*
Accuracy	88.3%	92.3%	94.7%	96.5%	97.3%
Elapsed time(seconds)	0.36	0.42	0.45	0.47	0.50

(b) Shader

	k'=k	*k'=2k*	*k'=3k*	*k'=4k*	*k'=5k*
Accuracy	95.4%	99.4%	99.8%	100%	100%
Elapsed time(seconds)	2.35	2.64	2.91	3.12	3.24

In the fourth experiment, we measured the precision and recall of the proposed method and compared with the search result based on the EMD using SIMPLIcity in order to verify the robustness of the proposed method. In this experiment, we set $k' = 2k$ since SIMPLIcity is very similar to Shader. As a result, in Table 6, for all k' values, the proposed method showed almost the same result as the EMD-based search. Such a good result of the proposed method is obtained by reducing the errors of the AEMD by the MCW elimination and the false dismissals by k'-NN search on the M-tree.

Table 6. Precision and recall of the proposed method and EMD-based search

(a) Precision

	k=10	*k=20*	*k=30*	*k=40*	*k=50*
Proposed method	88.2%	75.0%	66.1%	54.4%	44.5%
EMD-based search	88.2%	75.0%	66.1%	54.5%	44.8%

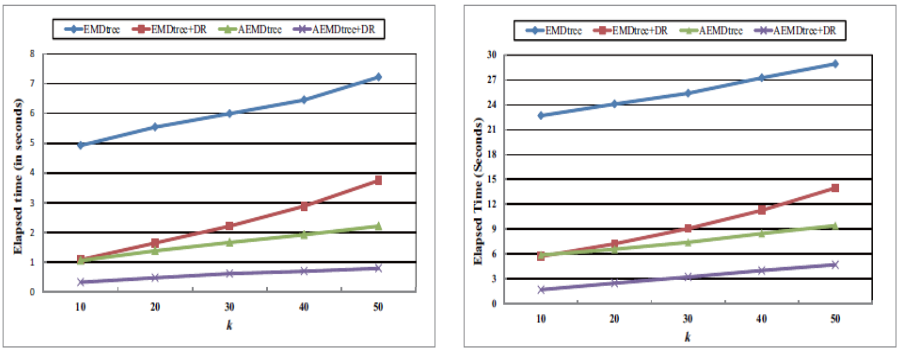
(b) Recall

	k=10	*k=20*	*k=30*	*k=40*	*k=50*
Proposed method	8.8%	15.0%	20.0%	27.2%	44.5%
EMD-based search	8.8%	15.0%	20.0%	27.3%	44.8%

As discussed in Section 4, the proposed method employs the three component techniques, namely the M-tree indexing, dimensionality-reduction, and the AEMD. In the fifth experiment, we examined the search performance due to various combinations of the three techniques in order to analyze the gain by each of them. The combinations in this experiment are summarized in Table 7.

Table 7. Combinations of component techniques

EMD_{tree}	Step 1: Build the M-tree with original histograms *H* Step 2: Perform *k*-NN_Search on the M-tree based on the EMD
$EMD_{tree+DR}$ (*k*-NN_EMD)	Step 1: Build the M-tree with dimensionality-reduced histograms *H'* Step 2: Perform *k*-NN_Search on the M-tree based on the EMD to find candidates Step 3: Refine *k*-NN by computing the EMD with the original histograms *H*
$AEMD_{tree}$	Step 1: Build the M-tree with the original histograms *H* Step 2: Perform *k'*-NN_Search on the M-tree based on the AEMD Step 3: Perform the post-processing based on the EMD on the original histograms *H*
$AEMD_{tree+DR}$ (our method)	Step 1: Build the M-tree with dimensionality-reduced histograms *H'* Step 2: Perform *k'*-NN_Search on the M-tree based on the AEMD to find candidates Step 3: Refine *k'*-NN by computing the AEMD with the original histograms *H* Step 4: Perform the post-processing based on the EMD on the original histograms *H*



(a) RETINA (b) Shader

Fig. 7. Comparison of search performance: the proposed AEMDtree+DR achieves the best performance

Figure 7 shows the results. Although the EMD_{tree} reduces the disk access overhead by using the M-tree, it shows the worst search performance because of the EMD computation overhead. The $EMD_{tree+DR}$, the *k*-NN_EMD in Figure 4, shows a better

performance than the EMD_{tree} because it reduces the EMD computation overhead in the M-tree search with the dimensionality-reduction. However, the overhead of computing the EMD is still very high in the post-processing step, since the number of candidates returned from the M-tree search is much larger than *k*. The $AEMD_{tree}$ performs the *k*-NN search on the M-tree based on the AEMD and thus achieves the faster search performance than the above two combinations. However, the $AEMD_{tree}$ suffers from the computational overhead due to the high-dimensional histograms in the M-tree search. On the other hand, the proposed method, the $AEMD_{tree+DR}$, reduces both the disk access overhead and the EMD computation overhead by combining the three component techniques and thus achieves the best search performance.

Table 8. Distribution of elapsed time of each step in three combinations (in seconds, on Shader)

(a) $EMD_{tree+DR}$

	M-tree search		*k*-NN refinement time	Total elapsed time
	Search time	I/O reads		
k=10	2.40	291.95	3.32	5.72
k=20	3.09	329.90	5.99	9.08
k=30	3.99	351.75	9.99	13.98

(b) $AEMD_{tree}$

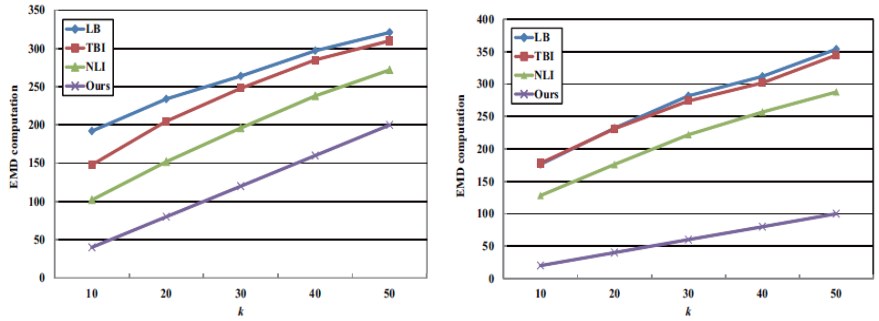
	M-tree search		Post-processing time	Total elapsed time
	Search time	I/O reads		
k=10	5.64	472.10	0.25	5.89
k=20	6.69	547.85	0.73	7.42
k=30	8.18	621.85	1.26	9.44

(c) $AEMD_{tree+DR}$

	M-tree search		*k'*-NN refinement time	Post-processing time	Total elapsed time
	Search time	I/O reads			
k=10	1.40	304.28	0.25	0.25	1.90
k=20	1.94	342.40	0.57	0.73	3.24
k=30	2.62	379.40	0.83	1.26	4.71

Table 8 shows the time (in the unit of seconds) elapsed in each step except the first M-tree building shown in Table 7 for three combinations $EMD_{tree+DR}$, $AEMD_{tree}$, and $AEMD_{tree+DR}$. The time is shown only for Shader, since RETINA shows very similar distribution of elapsed time.

As shown in the table, the $EMD_{tree+DR}$ completes the M-tree search quickly using the dimensionality-reduction. However, the total elapsed time is large due to the high EMD computation overhead in the refinement step. The $AEMD_{tree}$ searches for the k'-NN objects ($k' > k$) from the M-tree with the original high-dimensional histograms and thus has both the M-tree search time and I/O costs larger than the $EMD_{tree+DR}$. Nevertheless, the $AEMD_{tree}$ has the smaller elapsed time, since its post-processing needs to compute the EMD only for k' objects retrieved from the M-tree. The $AEMD_{tree+DR}$ has the smallest execution time even though it consists of one more k'-NN refinement step, which finds k'-NN objects from the candidates retrieved from the M-tree based on AEMD and thus is completed very quickly as shown in Table 8(c). It also needs to compute the EMD only for k' objects in the post-processing step, whose execution time is very short as in $AEMD_{tree}$.

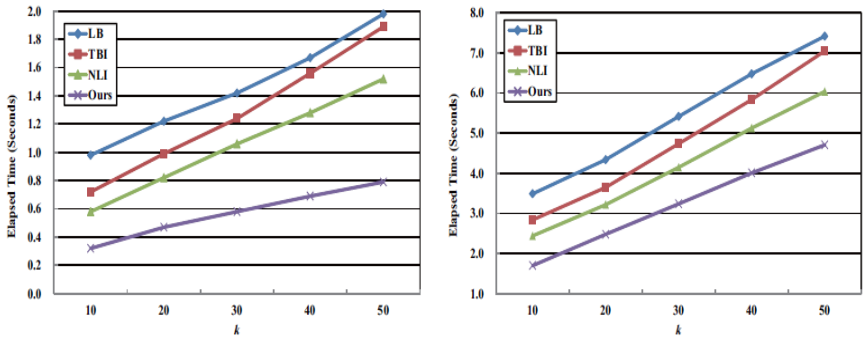


(a) RETINA (b) Shader

Fig. 8. Comparison of the number of EMD computations with respect to varying values of k: our method always has the smallest number of EMD computations

In the final experiment, we compared the number of EMD computations and the k-NN search time of the proposed method with the previous methods, namely LB [18], TBI [8], and NLI [12]. Figure 8 compares the number of EMD computations with respect to varying values of k. The proposed method shows the smallest number of EMD computations. Compared with LB, TBI, and NLI, our method reduces the EMD computations by up to 79.2%, 72.9%, and 60.7% on RETINA, and up to 88.7%, 88.6%, and 84.2% on Shader, respectively. The previous methods first search a set of candidates using the index or the lower-bounding function and then compute the EMD of each candidate to find the final k-NN. They perform many EMD computations, because the number of candidates is very large. On the contrary, the proposed method needs to compute the EMD only for k' candidates in the post-processing step. Therefore, the number of the EMD computations of our method is much smaller than the previous methods.

Figure 9 compares the total search time. Compared with LB, TBI, and NLI, our method reduces the search time by up to 67.3%, 55.5%, and 44.8% on RETINA and up to 51.7%, 40.1%, and 31.3% on Shader, respectively. LB shows the worst performance: it suffers from the scalability problem since it does not use an index and also needs many EMD computations as shown in Figure 7. TBI and the NLI show the better performance than LB, since they use the indexes. However, they require a considerable number of EMD computations as shown in Figure 7, which causes performance degradation. The proposed method reduces not only the disk access overhead using the index but also the number of EMD computations using the AEMD and thus has the better performance than the previous methods. We claim that our method should be recognized as more useful than the others especially for the CBIR on very large multimedia databases.



(a) RETINA (b) Shader

Fig 9. Comparison of search time with respect to varying values of k: our method always has the best performance than the others

6. Conclusion

In this paper, we proposed an approximate *k*-NN search method for the EMD-based CBIR. To perform the efficient *k*-NN search, both the disk access overhead and the EMD computational overhead should be reduced. The proposed method adopts the M-tree, a distance-based index structure, to reduce the disk access overhead. When building the M-tree, our method reduces the number of bins of the histograms using the dimensionality-reduction. After constructing the M-tree, it finds a small number of candidates from the disk by using the M-tree and performs the post-processing on them. The proposed method uses the approximate EMD in index retrieval and post-processing to reduce the computational overhead of the EMD. Also, we proposed the *k'*-NN search and the maximum common weight elimination to compensate the errors caused by the approximation. The extensive experiments reveal that our method achieves the significant improvement in terms of the number of the EMD computations and the *k*-

NN search time. The proposed method improves the performance of the previous method by up to 67.3% and only incurs 3.5% error. The retrieval results of the proposed method in real-world databases are almost the same as those of the original EMD, which indicates the errors of the proposed method hardly influence the quality of CBIR results. Given the fast processing time and small errors, our method can be used effectively in the CBIR for large databases.

Acknowledgements. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. 2017R1D1A1B03030969). This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2013-1-00881) supervised by the IITP (Institute for Information & communication Technology Promotion) and supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (No. NRF-2017M3C4A7083678).

A APPENDICES

Proof of Lemma 2:

Let the flow matrix $\widehat{\boldsymbol{F}} = [\hat{\boldsymbol{f}}_{\boldsymbol{ij}}]$ be the approximate minimum work of the AEMD. The AEMD can be defined as follows:

$$AEMD_D = \sum_{i=1}^{n}\sum_{j=1}^{n} \hat{f}_{ij} d_{ij}$$

The $AEMD_{reduced}$ with the reduction matrix R is defined as follows:

$$AEMD_{D'} = \sum_{i'=1}^{n'}\sum_{j'=1}^{n'} \hat{f}_{i'j'}\, d'_{i'j'}$$

$$\hat{f}_{i'j'} = \sum_{\{i|r_{ii'}=1\}}\sum_{\{j|r_{jj'}=1\}} \hat{f}_{ij}$$

$$d'_{i'j'} = \min\{d_{ij} | r_{ii'} = 1 \wedge r_{jj'} = 1\}$$

Based on these equations, we can infer the following equations:

$$AEMD_D = \sum_{i=1}^{n}\sum_{j=1}^{n} \hat{f}_{ij} d_{ij}$$

$$= \sum_{i'=1}^{n'}\sum_{j'=1}^{n'} \left(\sum_{\{i|r_{ii'}=1\}} \sum_{\{j|r_{jj'}=1\}} \hat{f}_{ij} d_{ij} \right) \tag{A.1}$$

$$\geq \sum_{i'=1}^{n'}\sum_{j'=1}^{n'} \left(\left(\sum_{\{i|r_{ii'}=1\}} \sum_{\{j|r_{jj'}=1\}} \hat{f}_{ij} \right) \times \min\{d_{ij} | r_{ii'} = 1 \wedge r_{jj'} = 1\} \right) \tag{A.2}$$

$$= \sum_{i'=1}^{n'}\sum_{j'=1}^{n'} \left(\left(\sum_{\{i|r_{ii'}=1\}} \sum_{\{j|r_{jj'}=1\}} \hat{f}_{ij} \right) \times d'_{i'j'} \right) \tag{A.3}$$

$$= \sum_{i'=1}^{n'}\sum_{j'=1}^{n'} \hat{f}_{i'j'} d'_{i'j'} = AEMD_{D'} \tag{A.4}$$

By using the reduction matrix, the $AEMD_D$ can be represented as Equation (A.1). If we replace d_{ij} by the minimum value that is satisfied $\{r_{ii'} = 1 \wedge r_{jj'} = 1\}$ as shown in Equation (A.2), the result is less than or equal to the $AEMD_D$. Since $d'_{i'j'}$ denotes the minimum value of the optimal reduced distance matrix D' (in Equation (A.3)), the $AEMD_{reduced}$ is always less than or equal to the $AEMD_D$ (in Equation (A.4)).

References

1. Y. Liu, G. Zhange, W. Ma.: A survey of content-based image retrieval with high-level semantics. Pattern Recognition, 40 (1): 262-282. (2007)
2. W. Zhou, H. Li, Q. Tian.: Recent Advance in Content-based Image Retrieval: A Literature Survey. eprint arXiv:1706.06064[cs.MM]. (2017)
3. M. Yasmin, S. Mohsin, M. Sharif.: Intelligent Image Retrieval Techniques: A Survey. Journal of applied research and technology, 12(1): 87-103. (2014)
4. A. Alzubi, A. Amira, N. Ramzan.: Semantic content-based image retrieval: A comprehensive study. Journal of Visual Communication and Image Representation, 32: 20–54. (2015)
5. E. Rashedi, H. Nezamabadi-pour, S. Saryazdi.: A simultaneous feature adaptation and feature selection method for content-based image retrieval systems. Knowledge-Based System, 39: 85-94. (2013)
6. E. Yildizer, A. Balci, T. Jarada, R. Alhajj.: Integrating wavelets with clustering and indexing for effective content-based image retrieval. Knowledge-Based Systems, 31: 55-66. (2012)
7. P. Ciaccia, M. Patella, P. Zezula.: M-tree: An efficient access method for similarity search in metric spaces. Proceedings of Very Large Data Bases, 426-435. (1997)

8. J. Xu, Z. Zhang, A. Tung, G. Yu.: Efficient and effective similarity search over probabilistic data based on earth mover's distance. Proceedings of International Conference on Very Large Data Bases,758-769. (2010)
9. I. Assent, A. Wenning, T. Seidl.: Approximate techniques for indexing the earth mover's distance in multimedia databases. Proceedings of IEEE International Conference on Data Engineering, 11. (2006)
10. H. Chen, et al.: A novel bankruptcy prediction model based on an adaptive fuzzy k-nearest neighbor method. Knowledge-Based Systems, 24 (8): 1348-1359. (2011)
11. T. Seidl, H. Kriegel.: Optimal multi-step k-nearest neighbor search. Proceedings of ACM International Conference on Management of Data, 154-165. (1998)
12. B. Ruttenberg, A. Singh.: Indexing the earth mover's distance using normal distributions, Proceeding of International Conference on Very Large Data Bases, 205-216. (2012)
13. I. Assent, M. Wichterich, T. Meisen, T. Seidl.: Efficient similarity search using the earth mover's distance for large multimedia databases. Proceedings of IEEE International Conference on Data Engineering, 307-316. (2008)
14. Y. Rubner, C. Tomasi, J. Guibas.: The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision, 40 (2): 99-121. (2000)
15. S. Shirdhonka, D. Jacobs.: Approximate earth mover's distance in linear time. Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 1-8. (2008)
16. M. Jang, S. Kim, C. Faloutsos, S. Park.: A linear-time approximation of the earth mover's distance. Proceedings of ACM International Conference on Information Knowledge Management, 505-514. (2011)
17. V. Ljosa, A. Bhattacharya.: Indexing spatially sensitive distance measures using multi-resolution lower-bounds. Proceedings of EDBT International Conference on Extending Database Technology, 865-883. (2006).
18. M. Wichterich, I. Assent, P. Kranen, T. Seidl.: Efficient emd-based similarity search in multimedia databases via flexible dimensionality reduction. Proceedings of ACM International Conference on Management of Data, 199-211. (2008)
19. B. Moon, et al.: Analysis of the clustering properties of the Hilbert space-filling curve. IEEE Transactions on Knowledge and Data Engineering, 13 (1): 124-141. (2001)
20. C. Papadimitrous, K. Steiglitz.: Combinatorial Optimization: Algorithms and Complexity. 1st ed. NewYork: Dover Publications. (1998)
21. Y. Tang, et al.: The earth mover's distance based similarity search at scale. Proceedings of the VLDB Endowment, 313-324. (2013)
22. N. Beckmann, H. Kriegel, R. Schneider, B. Seeger.: The r*-tree: An efficient and robust access method for points and rectangles. Proceedings of ACM International Conference on Management of Data, 322-331. (1990)
23. M. Jang, et al.: On extracting perception-based features for effective similar shader retrieval. Proceedings of IEEE International Conference on Computer Software and Applications, 103-107. (2011)
24. J. Wang, J. Li, G. Wiederhold.: Simplicity: Semantics-sensitive integrated matching for picture libraries. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23 (9): 947-963. (2001)
25. J. Shepperd, C. Schofield.: Estimating software project effort using analogies. IEEE Transactions on Software Engineering, 23 (11): 736-743. (1997)
26. D. C. G. Pedronette, R. S. Torres.: A correlation graph approach for unsupervised manifold learning in image retrieval tasks. Neurocomputing Joural, 208: 66-79. (2016)
27. S. Sun, Y. Li, W. Zhou, Q. Tian, H. Li.: Local residual similarity for image re-ranking. Information sciences, 471: 143-153. (2017)
28. J. Wu, Y. He, X. Qin, N. Zhao, Y. Sang.: Click-Boosted Graph Ranking for Image Retrieval. Computer Science and Information Systems, Vol. 14, No. 3, 629–641. (2017)

Min-Hee Jang earned the M.S. and Ph.D. degrees in electronics and computer engineering from Hanyang University, Korea, at 2006 and 2012, respectively. In 2013, he joined Carnegie Mellon University, Pittsburgh, USA, at the Computer Science Department as a postdoctoral researcher. He is currently working at Samsung Electronics as a software engineer and data analyst from the summer of 2014. His research interests include data mining, multimedia information retrieval, and social network analysis.

Sang-Wook Kim received the BS degree in Computer Engineering from Seoul National University, Seoul, Korea in 1989 and earned the MS and PhD degrees from Korea Advanced Science and Technology (KAIST) at 1991 and 1994, respectively. He is Professor at Department of Computer Science and Engineering, Hanyang University, Seoul, Korea. Professor Kim worked with Carnegie Mellon University and IBM Watson Research Center as a visiting scholar. He is an associate editor of Information Sciences. His research interests include data mining and databases.

Woong-Kee Loh received his B.S. (1991), M.S. (1993), and Ph.D. (2001) degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), South Korea. Currently, he is an associate professor in Dept. of Software, Gachon University, South Korea. He has served as a program and an organizing committee member in many international conferences including DaWaK 2007-08, CIKM 2009, PAKDD 2011-14, ICDE 2015, SSTD 2015, DASFAA 2017, and BigComp 2017-19. His research interests include massively parallel large-scale data mining.

Jung-Im Won received her M.S. and Ph.D. degrees in computer engineering from the Hallym University, Korea, in 1997 and 2004, respectively. Now she is a research professor in Hallym University in Korea. Her research interest includes database system, data mining and bioinformatics.

Received: October 10, 2018; Accepted: March 25, 2019

Computer Science and Information Systems 16(2):639–655 https://doi.org/10.2298/CSIS181015013C

Lexicon Based Chinese Language Sentiment Analysis Method

Jinyan Chen[1], Susanne Becken[1], and Bela Stantic[2]

[1] Griffith Institute for Tourism
Griffith University, Queensland, Australia
{Jinyan.Chen@griffithuni.edu.au
[2] Griffith Institute for Tourism
Griffith University, Queensland, Australia
s.becken@griffith.edu.au
[3] School of Information and Communication Technology
Griffith University, Queensland, Australia
B.Stantic@griffith.edu.au

Abstract. The growing number of social media users and vast volume of posts could provide valuable information about the sentiment toward different locations, services as well as people. Recent advances in Big Data analytics and natural language processing often means to automatically calculate sentiment in these posts. Sentiment analysis is challenging and computationally demanding task due to the volume of data, misspelling, emoticons as well as abbreviations. While significant work was directed toward the sentiment analysis of English text there is limited attention in literature toward the sentiment analytic of Chinese language. In this work we propose method to identify the sentiment in Chinese social media posts and to test our method we rely on posts sent by visitors of Great Barrier Reef by users of most popular Chinese social media platform Sina Weibo. We elaborate process of capturing of weibo posts, describe a creation of lexicon as well as develop and explain algorithm for sentiment calculation. In case study, related to sentiment toward the different GBR destinations, we demonstrate that the proposed method is effective in obtaining the information and is suitable to monitor visitors' opinion.

Keywords: Sentiment Analysis, Social Media, Natural Language Processing.

1. Introduction

Recent advances in computer science, technology and communication, in combination with advanced equipment and services, reinvented the channels through which people collect and generate information. The fundamental concept of generating data has changed. In the past, a small number of main sources have been generating data and all other actors have been consuming data. However, today all of us are both generating data and we are also consumers of this shared data. This is particularly evident in relation to social media platforms, which are attracting more and more users who talk about diverse topics. Despite concerns, related to privacy and credibility of posted information, this new method of obtaining relatively independent information about the quality of a product or service has benefits as it is limiting ability of businesses to control and influence customers. Feedback about a wide range of services and products can now be acquired from independent

reviews by consumers themselves. These types of reviews are known as user-generated content, and they have been found to play an important role in customers future behaviors due to the electronic word-of-mouth [38].

The growing role of social media is attracting increasing research interest. Social media plays significant role in many aspects of life including retails, politics, tourism, and decision-making behavior. A large number of users actively engage with social media, for example, Twitter has 313 million monthly active users worldwide[27]. Furthermore, over 1.94 billion people monthly use Facebook [39] and 700 Million users use Instagram [15], and the Chinese Social Media platform Sina Weibo has over 400 million active users [11]. People use social media to post stories from their daily life, and they are particularly likely to share impressions or emotions related to their travel experience. For the tourism industry, it is therefore possible to examine social media and understand what visitors share and how they perceive destinations, attractions and products.

The Great Barrier Reef (GBR) is the world's largest coral reef system stretching over 2,600 kilometers along the coast of Queensland and it attracts over two million visitors each year from all around the world [12]. Significant proportion of GBR visitors are from China. While Chinese visitors are very active users of social media they typically use Chinese platforms such as Sina Weibo rather than those commonly used by other English speaking visitors. To take advantage of this huge number of users and media posts on Weibo in this work we decided to capture those posts which specifically talk about the GBR.

Advances in Big Data analytic and natural language processing provide the opportunity to analyse visitor experience and their sentiment toward certain services or places [26]. Sentiment analysis is a method which can be used for analyzing social media content. It basically converts social media post text into quantitative data, whereby it can extract information about special events and to identify patterns. Sentiment analysis is a challenging and computationally demanding task not just because of vast volume of data but also due to the common grammatical errors and misspelling, slang and abbreviations. Additionally, social media post can contain sentiment emoticons that carry valuable information for calculating the sentiment scores and should not be discarded.

Social media posts have been harnessed for different purposes including monitoring environmental changes [4], [5] as well as sentiment analysis in tourism [19], [22], [37]. In most cases concept relied on sentiment derived from short text of social media posts and analytics of posts' meta data along other available online or scientific data. Different statistical and machine learning methods have been used, such as Support Vector Machines (SVM) and Naive Bayes. Recent literature also addresses the issues with regard to trust and reputation measures in social network systems, especially in presence of thematic social groups [10].

There are many sentiment analysis methods for English text presented in literature such as Valence Aware Dictionary for Sentiment Reasoning (VADER) approach, which is purposely developed for sentiment analysis of short text found in social media posts [14]. VADER relies on dictionary but also has set of rules, which takes into consideration punctuation, emoticons, and many other heuristics to compute sentiment polarity. Dictionary contains items with associated sentiment intensities which are annotated by humans. For instance, a dictionary may contain words, such as excellent, better, good, bad, worse, terrible, with their respective sentiment intensity and polarity. While the sentiment analysis

is matured for English language there are limited work for Chinese language sentiment analysis, and was mostly directed toward lexicon creation.

Majority of proposed sentiment analysis methods for Chinese language are machine learning based [41], for example, Xu and colleagues looked into to classify sentiments of microblogs from Sina Weibo. They relied on labeled emoticons as a training corpus and built a fast Bayesian classifier based on assumption that both smiley and emoticons are strongly related with typical sentiment words and are viewed as convincing indicators of emotions [36]. Authors of [21] claimed that they improved sentiment analysis by identifying features with SVM as a learning engine, which they named global optimization-based sentiment analysis approach. However, the authors pointed out that if the parameters are not selected well the result will not be accurate and that the sentiment feature subset choice influences appropriate kernel parameters, and vice versa. On the other hand authors of [41] relied on lexicon from National Taiwan University Sentiment Dictionary (NTUSD) [32], which has both traditional Chinese and simplified Chinese characters. NTUSD contains 2810 positive words (which are assigned +1 as sentiment intensity) and 8276 negative words (assigned -1 as sentiment intensity). Also [33] adopted NTUSD as lexicon to calculate a sentiment score to monitor the public opinion and forecast election [6], however, they also rely on lexicon which has limited number of words and additionally has only two levels of sentiment intensity +1 or -1, associated with positive and negative words.

To overcome above mentioned shortcomings of existing approaches, in this work we propose and test method to identify sentiment in Chinese social media posts and we rely on most popular Chinese social media Sina Weibo [25]. We developed method to crawl the web and collect Weibo posts that mentioned specific words (in this case word "Great Barrier Reef" in Chinese language). We elaborate process of capturing, managing, we describe creation of comprehensive Chinese lexicon with sentiment intensity and we also propose algorithm for sentiment calculation, which takes into consideration length of the post as well as number of matching words with lexicon. Additionally as a proof of concept, we provide sample of additional information, which can be extracted from the social media post meta data; such as where from Chinese people who have interest and comment on GBR originate from as well as what is average sentiment depending on province poster originate from.

2. Background

Social media has influenced the way people search information and make decisions. Tourism organizations and destination marketing organizations are aware of ongoing trends and thus try to explore the opportunities to use tourist-generated content for their own marketing and include it as an integral part of their branding and positioning.

2.1. Social Media Data

The emergence of user-generated content (UGC) on the Internet in mid-2000s has provided a new source of data and enabled millions of tourists to exchange content on popular platforms for mutual benefit, such as social networking (e.g., Facebook), micro-blog (e.g., Weibo, Twitter), multimedia sharing (e.g. YouTube), location sharing (FourSquare), and review forums (TripAdvisor).

Social media data has many forms, Figures 1 and 2 show several samples only to demonstrate the diversity, while below we list and highlight main characteristics of social media data:

1. It generate massive volume of data.
2. Data are generated with unprecedented speed.
3. Data are complex and high-dimensional, attribute-value data such as text, comments, and other meta data about the poster, Figure 1.
4. Data exist in different forms: text, emotions, images, videos, etc. See Figure2
5. Additionally, data can be noisy as well as incomplete and contain a lot os misspelling, slang and abbreviations.


```
"_id" : ObjectId("5b337062af543679e3c87d03"),
"reposts_count" : 0,
"text" : ": 六月马拉松 黄金海岸在等你
"geo_enabled" : 1,
"geo" : {
        "coordinate" : "-35.008038,138.571808",
        "location" : $$$
},
"id" : "2202125923M_G8ylSwhnP",
"likes_count" : 0,
"imgurl_list" : [
        "$$$"
],
"created_at" : "2018-03-22 15:01",
"userid" : NumberLong("2202125923"),
"mid" : "$$",
"terminal" : "none",
"source" : "$$",
"comments_count" : 3,
"userinfo" : {
        "gender" : "女",
        "region" : "海外 澳大利亚",
        "name" : " $$$",
        "birthdate" : "01-01"
}
```


Fig. 1. Sample Weibo data indicates complex high-dimensionality of data

Social media platforms including Twitter, Flickr, and Sina Weibo also offer possibility of geo-referencing the content shared by users. This enables opportunity to trace social media users and compare findings with other sources of data.

Sina Weibo is a Chinese micrologic website launched by Sina Corporation in 2009, it is similar to Twitter but from 2016 with some flexibility with regard to the length of the post [3]. Sina Weibo is the first and the most popular Chinese social media platform [17], it has more than 411 million monthly active users in first quarter of 2018 [28].

Fig. 2. Several samples of data formats.

2.2. Sentiment Analysis

Sentiment analysis or opinion mining relates to the study of opinions, attitudes, and emotions toward entities, individuals, issues or events. Sentiment analysis dates back to 1970s and is conceptually grounded in the work of Osgood and his associates on content analysis of peoples judgments by evaluating text [24]. Osgood and colleagues distinguish between three dimensions of meaning: *Evaluation, Potency,* and *Activity*. Assessment good-bad or favorable-unfavorable are along the *Evaluation* dimension of meaning, while intensity of these evaluations such as strong/weak or powerful/powerless represent *Potency*, and fast/slow or active/passive comprise the *Activity* dimension.

Today we talk about sentiment analysis as a process of computationally identifying and categorizing opinions in order to determine the writer's attitude toward a particular issue. This can be achieved by employing computer-based natural language processing which aims to detect sentiment by relying on Artificial Intelligence system that would be able to reason about emotion [18].

A comprehensive sentiment analysis also considers meta data such as, who provided the information and at what time. Literature elaborates methods of sentiment analysis which in general falls into one of three categories [1]:

– **Machine learning**: Machine learning approaches involve creating a model by using human annotated data. Mostly supervised machine learning approaches have been mentioned in literature and these methods are composed of pre-processing, feature extraction, learning, and classification steps.

- **Dictionary-based**: Also refereed as Lexicon based, as there is associated polarity and weight on each word in dictionary. These systems mainly rely on the use of dictionaries containing comprehensive sentiment lexicons and sets of fine-tuned rules. A sentiment dictionary can be created either by humans (manually), by machine (automatically) or by humans and machine (semi-automatically).
- **Hybrid approaches**: These methods combine dictionary-based and machine learning-based approaches and they work in parallel to compute sentiment polarities.

As mentioned earlier, typical representative of Dictionary based methods is VADER, it is suited for sentiment analysis of short text [14]. Clear advantage of dictionary based methods for sentiment analysis is that there is no need for annotation of the text for training. Another advantage is possibility to create domain specific dictionaries which ensure higher accuracy of calculated sentiment scores. But there is still need to create dictionary and annotate polarity of words, however, it needs to be done only once. This is obviously less costly considering that the annotation for supervised machine learning method needs to be undertaken for each new context.

In hybrid approaches both dictionary and machine learning are used to independently compute sentiment and then individual results are combined to provide sentiment intensity and polarity [16]. Specific hybrid model which is using keywords and Naive Bayes algorithm has been recommended to calculate sentiment polarities of social media tweets [8].

3. Methodology

A sentiment analysis process is composed of several independent steps as it can be seen in Figure 3. It starts with data collections which in case of social media data most often relies on utilization of dedicated available Application Programming Interface (API). In cases when purpose built API is not available web crawling is needed. Crawling initially involves an identification of a data source, for example, a commercial website or a social media network. Dedicated web crawling code needs to be developed and used to collect data from these sources. Considering that huge amount of post have been generated by users there is need to filter and acquire only relevant posts, most often filtering is related to particular geographic area or to particular keywords. After initial cleaning, which discards data that do not contain full records, data are saved in database.

Once data have been stored in a database, it is possible, for example, to calculate the sentiment from content of the text within a set of data fields in particular posts. In addition to the actual text of the post additional meta data fields, if publicly available can also be captured. These field can be used to analyze locations where from posts have been sent or where users originate from, which is based on place users indicated as their location when they created accounts.

Due to the limited options with public Weibo API, we developed in Python programming language dedicated method to crawl the Weibo website and to collect relevant social media posts. Considering that data is in Java Script Object Notation (JSON) format we stored it locally in Mongo NoSQL database, which was chosen because it has been shown that the relational databases are struggling in handling large amount of unstructured data [26]. NoSQl database is able to store and efficiently access diversity of unstructured data including text, emotions and media files.

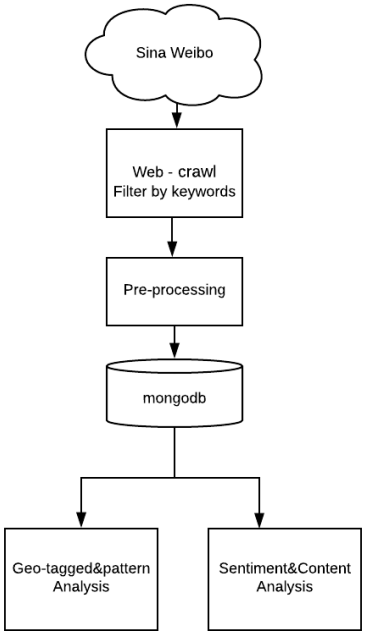


Fig. 3. Global Sentiment analysis process of Weibo data

The following procedure was implemented: we extracted only posts that contained the keywords ”Great Barrier Reef”, in Chinese language. Extraction process is shown in Figure 3. We identified that about 15% of posts have exact geo location (Latitude, Longitude), which can help in analysis of meta data and for example provide information about visited places. In addition, the analysis involves assessment of the ’emotional’ tone of the text; which will be calculated with method proposed in this paper.

Our experiments were conducted on in-house Big Data cluster running Hadoop (2.6.0) and MongoDB (3.2.9).

3.1. Creation of Chinese Lexicon

Considering that there is limited work on sentiment for Chinese text, it is also reflected in shortage as well as quality of available Chinese language lexicons. When looking into available dictionaries we have identified that despite there are some duplication of words these dictionaries complement each other. However biggest disadvantage was that they had no intensity of sentiment. Only lexicon from Bo Yuan, Tinghua University [7] has intensity associated with specific words, however, it has limited number of words. Other approach such as HowNet [40] haS intensity associated with different dictionaries such as ’most’, ’more’, etc. Also, dictionaries from Dalian University has a simple label for positive and negative polarity. All other dictionaries that we found are formed and grouped by positive or negative words.

In creating our lexicon we considered the following dictionaries:

- Chinese Sentiment Word Weight Table from BoYuan in Tinghua University: It contains 23,419 phrases with a positive or negative weight [7].
- How Net: has 17,887 phrases which are divided into 6 groups based on the phrases emotional tendency, which are ”Positive Evaluation”, ”Negative Evaluation”, ”Positive Emotion”, ”Negative Emotion”, ”perception” and ”Adverb of degree” [40].
- Chinese emotional vocabulary from Dalian University of Technology Information Research Laboratory. It contains 22,012 phrases [35].
- National Taiwan University Sentiment Dictionary - NTUSD: has both Simplified and Traditional Chinese Character. It has 2,810 positive words and 8,276 Negative words [31], [32].
- Tsinghua University Positive and Negative Dictionary which contains 4,468 negative words and 5,567 positive words [20].

Apart of being limited in content and number of words these dictionaries have shortcoming as there is no sentiment intensity measures of individual words. Weight has been mostly addressed by creating dedicated dictionaries for specific words which are grouped based on association with: most, very, more, half, etc. This causes lower accuracy of sentiment score and also requires dedicated code in programming for calculation of sentiment, which significantly slows down the sentiment calculation.

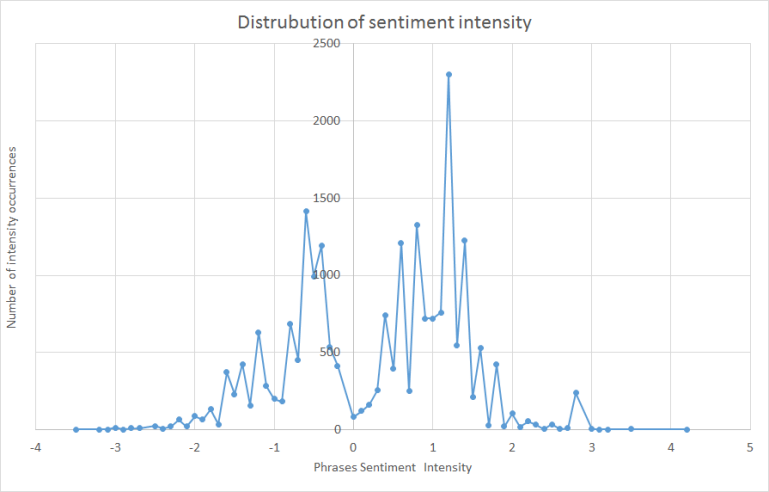


Fig. 4. Distribution of sentiment intensity of BoYuan Tinghua University lexicon

Therefore we created our own lexicon which combined existing dictionaries and added weight intensity in accordance with meaning. In absence of human annotation, for example we associated weight 3 times for words associated with *most* while 2 times for *more*, etc. Also, we assigned weight +1 to words in positive dictionaries add -1 for words in negative dictionary. Words which existed in lexicons and had sentiment intensity validated by humans we retained, such as from HowNet lexicon [9], [40].

Result was comprehensive lexicon for Chinese language with over 40,000 terms with associated weighting. We realize that the certain words still have only weight +1 or -1, however, it is at this stage sufficient and more accurate than any existing method and any existing lexicon. To further improve accuracy it is possible to engage people to do the annotation of words sentiment weight, however, considering the fact that there are over 40,000 words and every individual could have different sentiment weight it is required to survey several people for same terms, therefore this is lengthy and expensive task.

In Figure 4 is shown the distribution of BoYuan Tinghua University lexicon and Figure 5 presents the distribution of lexicon we compiled as part of this work, which is the combination of several existing dictionaries.

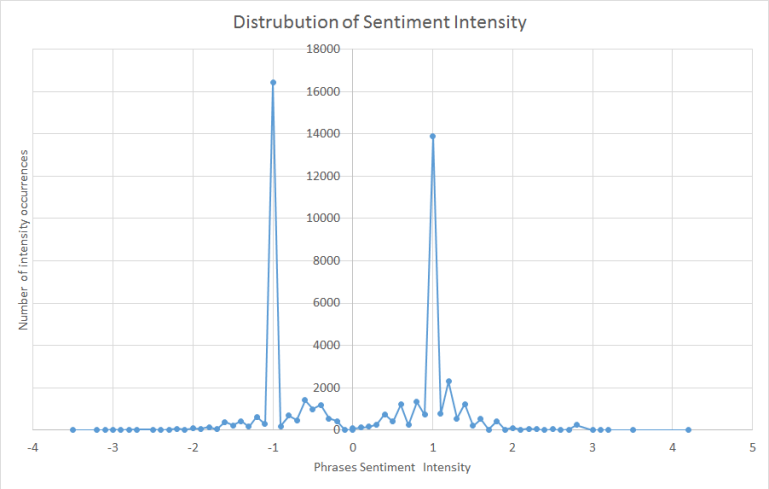


Fig. 5. Distribution of sentiment intensity of our Lexicon

3.2. Sentiment Analysis of Chinese Short Text

As indicated in Section 2 there are many sentiment analysis methods proposed for English language and they are mostly lexicon based. With regard to sentiment analysis of short Chinese language text there is limited work and process itself is more complicated when compared to the sentiment analysis of English language, because text also requires segmentation into meaningful terms. In Chinese language there are no spaces between characters and specific meaning is often defined by combination of characters or words. Encouraged by work of our Big data group on sentiment analysis for English language we decided to follow same direction and propose lexicon based sentiment analysis method for Chinese language. In Figure 6 we show concept of sentiment analysis of Chinese text that has been proposed and developed in this work.

Calculation of sentiment was done according to the Algorithm 1. Once *WEIBO* Chinese social media posts have been collected and stored in Mongodb databases in JSON

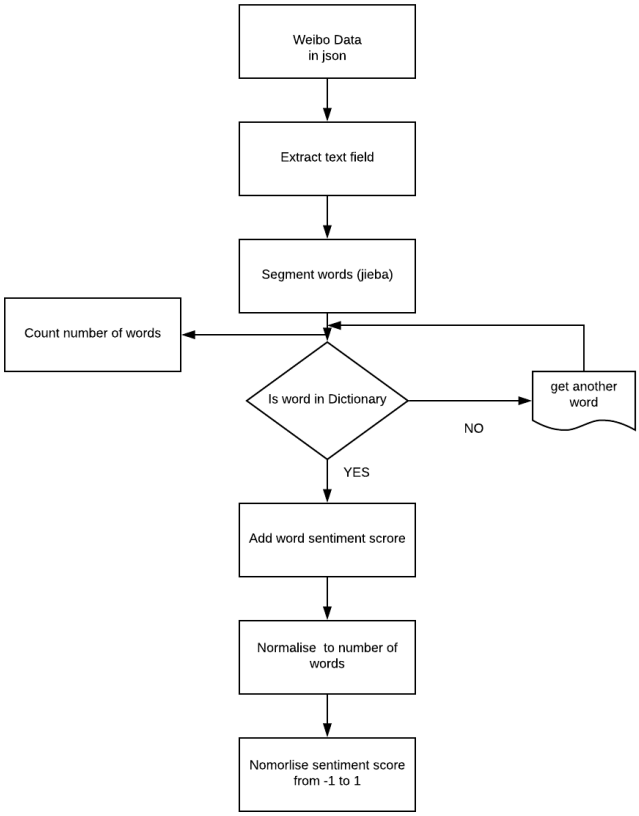


Fig. 6. Sentiment analysis concept

format, with regard to sentiment calculation of posts first step is to only consider the content of the posted text within the post *TEXT* and encode text with UTF-8 encoding *D*, suitable for Chinese language. Individual encoded text *d* is then considered for sentiment calculation. In the next step, considering that there is no space to separate characters and words in Chinese language, there is need to do the segmentation to meaningful words. In Chinese language, one character usually does not have meaning, which is a challenge itself to segment text written in Chinese into meaningful terms. For example, in English ”I love traveling”, ”traveling” is one word , however, in Chinese language, it needs two characters ”lv” and ”you” to have the meaning of ”traveling”. Additionally, as there is no space to separate words in a sentence, if we need to find ”traveling” in Chinese language it is required to take into consideration both ”lv” and ”you” in order to have the meaning of ”traveling”. Segmentation of Chinese language is research topic itself, and it is obviously outside of content of this work, it attracted significant attention in literature [13], [29], [2], [30], [23].

In this work on developing sentiment analysis of Chinese short text we accepted and followed *jieba* segmentation method [29]. Apart of being widely used *jieba* has advantage

because it has libraries for python, which we could simple plug into python code for sentiment analysis. *Jieba* can search the maximum probability path and most probable combination based on the word frequency. It supports three word-segmentation models (accurate mode, full mode and search engine mode), can process the traditional Chinese word segmentation, and supports custom dictionaries [34]. *Jieba* segmentation libraries files are publicly accessible at [29].

With regard to Algorithm 1, all segments from posts (*SEG*) are lopped and segments (*seg)* are matched with lexicon (*LEX)*. If segment exist in lexicon associated intensity (*wordsentiment*) is obtained and added to the Sentence sentiment ($sen\_sentiment$). Because post in Weibo can be short or very long, we count number of segments (*words*) as well as number of matching segments with lexicon (*dictwords*), which we take into consideration when calculating sentiment of individual post. This is required because there are more positive than negative words in Lexicon and therefore it is more likely that the lengthy posts have more matching words with lexicon and therefore result in higher positive sentiment if adjustment was not performed.

Algorithm 1: SENTIMENT CALCULATION OF SHORT CHINESE TEXT

Input: Weibo posts in JSON format
Output: Text and Sentiment scores
WEIBO = Input Weibo posts in JSON format
LEX = Import Lexicon with sentiment intensity
$TEXT = Extract\_only\_text\_field\_from\_post(WEIBO)$
$D = encode\_with\_UTF8(TEXT)$
$sen\_sentiment = 0$
for $d \in D$ **do**
  $words = 0$
  $dictwords = 0$
  $sentiment = 0$
  $SEG = Perform\_Segmentation\_of\_text\_with\_jieba\ (d)$
  for $seg \in SEG$ **do**
    $words = words + 1$
    if $seg \in LEX$ **then**
      $wordsentiment = LEX(seg)$
      $sen\_sentiment = sen\_sentiment + wordsentiment$
      $dictwords = dictwords + 1$
  $adjusted\_sent = sentiment * dictwords\ /\ words$
  $normalize\_to\_one = adjusted\_sent\ /\ \sqrt{adjusted\_sent^2 + 2}$
  $R = append(d,\ normalize\_to\_one)$
  return R

When all segments from individual post are taken into consideration at first adjusted sentiment ($adjusted\_sent$) was calculated with the following equation:

$$adjusted\_sent = sentiment * dictwords\ /\ words$$

where $dictwords$ represent number of words in post which has been matched with lexicon and $words$ is total number of words in post (to avoid influence of the length of the post to sentence sentiment score).

Finally post sentiment is normalized to 1 to ensure that the sentiment is always between -1, for negative, and +1 for positive post. The following equation is used to normalize sentiment to 1:

$$normalize\_to\_one \; = \; adjusted\_sent \; / \; \sqrt{adjusted\_sent^2 \; + \; 2}$$

Normalized sentiment score is stored in database along the associated text and loop continue to the next post until all posts are finished.

4. Experimental Evaluation

In our experimental evaluation we considered Sina Weibo posts posted in 2016 and 2017 that mentioned 'Great Barrier Reef', in Chinese language. A total of 24,308 relevant posts have been captured. As a demonstration of sentiment calculation we relied on our previous experience on Great Barrier Reef project [4], and selected several relevant keywords including key locations (Townswille, Cairns, etc), food (Seafood), hotel and some relevant activities such as 'snorkeling' and calculated overall sentiment for these keywords. In Table 1 we provide some sentiment scores calculated by method proposed in this work. It can be seen, for example, that all related posts are positive in these 2 years, however, the overall sentiment about the travel destinations was increasing, while only seafood sentiment score dropped from 0.4474 to 0.4043.

Table 1. Sentiment Score of GBR related posts

Key words	SentimentScore2016	SentimentScore2017
Cairns	0.3828	0.4042
Townswille	0.385	0.4484
Whitehaven Beach	0.3713	0.423
Whitsunday	0.3626	0.3473
Hamilton Island	0.354	0.343
Green Island	0.4398	0.4406
Heart Reef	0.481	0.4747
Hotel	0.3261	0.4314
Seafood	0.4474	0.4043
Snorkeling	0.3551	0.3525
Coral	0.3326	0.3766
Heart Reef	0.481	0.474
Fish	0.250	0.328

Considering that apart of text social media posts can contain other relevant metadata we investigated what additional information metadata can provide. We noticed that out of 24,308 captured posts, almost all users (99%) provided their location at the time of registration, which could be good indication where the users are coming from. Figure 7

shows a heat map to illustrate where in China the Weibo users who talked about the GBR originate from at the time of registration of their accounts. Based on this details we could identify number of users from different provinces, for example Beijing is top ranked (a total of 5,091 users) that mentioned Great Barrier Reef, followed by Guangzhou Province (2,389) and Shanghai (1,456). It is important to mention that this analysis result correlates well with the results released by Queensland Tourism Industry Council which announced that "Markets in Chinas first-tier cities (Beijing, Shanghai, Guangzhou and Shenzhen"). It has been also previously demonstrated that the social media data correlates well with scientific observations [5]. Therefore, considering also observation in this work related to first-tier cities it is evident that the metadata can be used to gain valuable information. It is interesting to note that there are posts from all provinces in China, indicating that the GBR is very popular in Chinese social media.

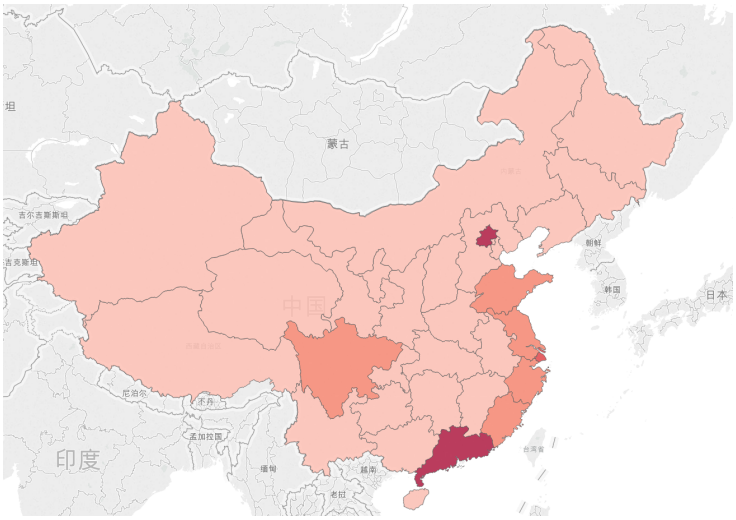


Fig. 7. Heat map of user locations that mentioned GBR from 2016 to 2017

Another analysis, shown in Figure 8, presents average sentiment in all posts that mentioned GBR depending on province where user opened their accounts. It is possible to see that some provinces have higher or lower sentiment, which could be investigated and found the reason for hihger or lower sentiment toward the GBR and findings can be used for change in marketing strategy. This is another sample how meta data can be used.

Also, we noticed that about 15% od posts have exact geographic locations (longitude, latitude) of posts, which can be used to identifythe points of interest.

Figure 9 shows distribution of posts in 2016. It is interesting to see that two peak times of posting about GBR are February and September with increasing trend. In the Chinese visitor satisfaction report says "Chinese holiday tourists arriving in Australia for leisure purpose have featured high seasonality. Australias summer tends to be the peak season for Chinese tourists in order to avoid freezing winter. The peak season lasts for four months

from November to February, covering several important holidays such as Christmas and New Year's Eve. Our finding correlates with this observation.

Fig. 8. Heat map of average sentiment in posts that mentioned GBR, depending on where users originate from

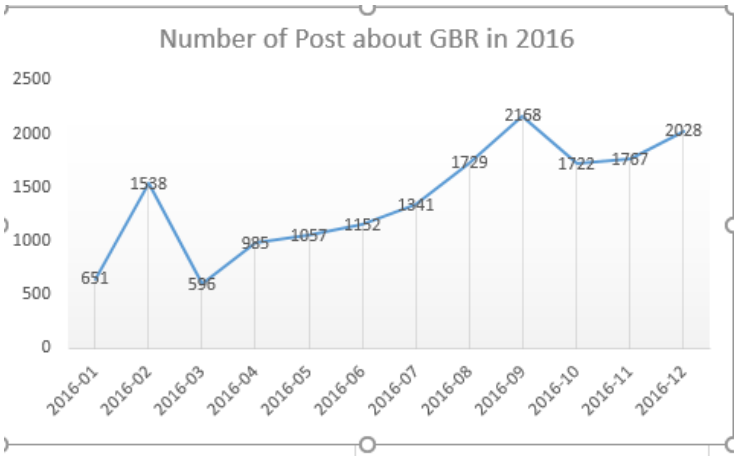


Fig. 9. Number of posts per month in 2016

5. Conclusions

Consumer perception assessments often rely on existing approaches to data collection such as surveys and opinion polls. However, these methods have a range of limitations both in terms of sample size and bias. There is also risk that the traditional methods are unable to capture opinions and behaviors due to relatively small sample size, as sample size is limited due to the cost of survey. To overcome these limitations, we proposed to tap into available social media post and perform Big Data analytics. Due to the fact that there are many Chinese tourist in Australia, specifically in area of Great Barrier Reef, we decided to gain insight into tourists with Chinese background by capturing and calculating sentiment of Sina Weibo Chinese social media posts.

Despite sentiment analysis is well researched and matured area for English language there is limited attention for Chinese language and is mostly concentrated to lexicon creation, which itself are small and do not have word sentiment intensities. To overcome shortcomings of existing approaches, in this work we proposed and tested method to identify sentiment in Chinese social media posts. We developed method to crawl the web and specifically collected Weibo posts that mentioned word Great Barrier Reef both in Chinese language and English language. We also elaborated process of capturing, managing, described creation of comprehensive Chinese lexicon with sentiment intensity and proposed algorithm for sentiment calculation. In contrast to all other existing methods proposed method takes into consideration length of the text as well as number of identified words in lexicon. To the best of our knowledge this is the very first method which avoid bias because there are more negative than positive words in lexicon and therefore longer posts tend to be more negative, due to the fact that in longer posts there are more likely to have more matching words in lexicon.

Additionally we provided samples of other valuable information, as a proof of concept, which can be extracted from social media posts meta data; such as where from Chinese people who have interest and comment on GBR originate from as well as what is the average sentiment depending on locations users indicated as their place of residence. In case study related to sentiment toward the different GBR destinations we demonstrated that the proposed method is effective to obtain information and to monitor visitors' opinion.

As of future work to further improve accuracy of sentiment analysis a more comprehensive lexicon is needed, especially a lexicon that can include words that people like to use in social media, as well as emoji, need to be taken into consideration.

References

1. Alaei, A.R., Becken, S., Stantic, B.: Sentiment analysis in tourism: Capitalizing on big data. Journal of Travel Research 58(2), 175–191 (2017)
2. Ansjsun: `http://www.cnblogs.com/en-heng/p/6274881.html`
3. BBC: Sina weibo ends 140-character limit ahead of twitter (2016), https://www.bbc.com/news/technology-35361157
4. Becken, S., Stantic, B., Chen, J., Alaei, A., Connolly, R.: Monitoring the environment and human sentiment on the great barrier reef: Assessing the potential of collective sensing. In: Journal of Environmental Management. vol. 203, pp. 87–97 (2017)

5. Chen, J., Wang, S., Stantic, B.: Connecting social media data with observed hybrid data for environment monitoring. In: Proceedings of the 11th International Symposium on Intelligent Distributed Computing - IDC 2017. pp. 125–135 (2017)
6. Chen, S., Ding, Y., Xie, Z., Liu, S., Ding, H.: Chinese weibo sentiment analysis based on character embedding with dual-channel convolutional neural network. In: 2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA). pp. 107–111. IEEE (2018)
7. Chinese Sentiment Word Weight Table, BoYuan, T.U.: https://github.com/bung87/bixin/tree/master/dictionaries
8. Claster, W.B., Cooper, M., Sallis, P.: Thailand–tourism and conflict: Modeling sentiment from twitter tweets using naïve bayes and unsupervised artificial neural nets. In: Computational Intelligence, Modelling and Simulation (CIMSiM), 2010 Second International Conference on. pp. 89–94. IEEE (2010)
9. Dong, P.Z.: Hownet (2000), http://www.keenage.com/html/e_index.html
10. Duong, C.T., Nguyen, Q.V.H., Wang, S., Stantic, B.: Provenance-based rumor detection. In: 28th Australasian Database Conference - ADC. pp. 125–137 (2017)
11. Fan, B.: Weibo data report in 2016 (2016), http://data.weibo.com/report/reportDetail?id=346
12. Greatbarrierreef: www.greatbarrierreef.org, http://www.greatbarrierreef.org/about-the-reef/great-barrier-reef-facts/
13. Hua-ping, Z., Qun, L.: Ictclas (2013)
14. Hutto, C., Gilbert, E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the 8th International AAAI Conference on Weblogs and Social Media (2014)
15. instagram: instagram (26 April,2017), http://blog.instagram.com/post/160011713372/170426-700million
16. Kasper, W., Vela, M.: Sentiment analysis for hotel reviews. In: Computational linguistics-applications conference. vol. 231527, pp. 45–52 (2011)
17. Kim, S.E., Lee, K.Y., Shin, S.I., Yang, S.B.: Effects of tourism information quality in social media on destination image formation: The case of sina weibo. Information & Management 54(6), 687–702 (2017)
18. Kirilenko, A.P., Stepchenkova, S.O., Kim, H., Li, X.: Automated sentiment analysis in tourism: Comparison of approaches. Journal of Travel Research p. 0047287517729757 (2017)
19. Leung, D., Law, R., Van Hoof, H., Buhalis, D.: Social media in tourism and hospitality: A literature review. Journal of Travel & Tourism Marketing 30(1-2), 3–22 (2013)
20. Li, J.: Chinese positive and negtive lecxion (22 January,2011), http://nlp.csai.tsinghua.edu.cn/site2/index.php/resources/13-v10%20Seeaustralia,%202017
21. Li, X., Li, J., Wu, Y.: A global optimization approach to multi-polarity sentiment analysis. PloS one 10(4), e0124672 (2015)
22. Liu, Z., Shan, J., Balet, N.G., Fang, G.: Semantic social media analysis of chinese tourists in switzerland. Information Technology & Tourism 17(2), 183–202 (2017)
23. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The stanford corenlp natural language processing toolkit. In: Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations. pp. 55–60 (2014)
24. Osgood, C.E.: The nature and measurement of meaning. Psychological bulletin 49(3), 197 (1952)
25. Sina: (2009-2018), https://weibo.com
26. Stantic, B., Pokornỳ, J.: Opportunities in big data management and processing. Databases and Information Systems 270, 15–26 (2014)
27. statista: Number of monthly active twitter users worldwide from 1st quarter 2010 to 4th quarter 2017 (in millions) (28 January,2014), https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

28. statista.com: Number of monthly active users (2018), https://www.statista.com/topics/1164/social-networks/
29. Sun, J.: jieba chinese word segmentation tool. available at https://github.com/fxsjy/jieba (2012)
30. Sun, M., Chen, X., Zhang, K., Guo, Z., Liu, Z.: Thulac: An efficient lexical analyzer for chinese. Tech. rep., Technical Report (2016)
31. University, N.T.: Http://academiasinicanlplab.github.io/
32. University, N.T.: Ntusd: National taiwan university semantic dictionary (22 January,2011), https://rdrr.io/rforge/tmcn/man/NTUSD.html
33. Wang, M.H., Lei, C.L.: Boosting election prediction accuracy by crowd wisdom on social forums. In: Consumer Communications & Networking Conference (CCNC), 2016 13th IEEE Annual. pp. 348–353. IEEE (2016)
34. Xiao, K., Zhang, Z., Wu, J.: Chinese text sentiment analysis based on improved convolutional neural networks. In: Software Engineering and Service Science (ICSESS), 2016 7th IEEE International Conference on. pp. 922–926. IEEE (2016)
35. Xu, L., Lin, H., Pan, Y., Ren, H., Chen, J.: Chinese emotional vocabulary. ???? 27(2), 180–185 (2008)
36. Xu, Y., Liu, Z., Zhao, J., Su, C.: Aweibo sentiments and stock return: A time-frequency view. PloS one 12(7), e0180723 (2017)
37. Yin, F., Zhang, B., Su, P., Chai, J.: Research on the text sentiment classification about the social hot events on weibo. In: IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC). pp. 1537–1541 (2016)
38. Zeng, B., Gerritsen, R.: What do we know about social media in tourism? a review. Tourism Management Perspectives 10, 27–36 (2014)
39. zephoria: Valuable facebook statistics (2017), hhttps://zephoria.com/top-15-valuable-facebook-statistics/
40. Zhendong, D., Qiang, D.: Hownet And The Computation Of Meaning. World Scientific (2006)
41. Zhuo, S., Wu, X., Luo, X.: Chinese text sentiment analysis based on fuzzy semantic model. In: Cognitive Informatics & Cognitive Computing (ICCI* CC), 2014 IEEE 13th International Conference on. pp. 535–540. IEEE (2014)

Jinyan Chen is a doctoral student from the Department of Tourism, Sport and Hotel Management at Griffith University. Her research interest is related to Big Data Analysis in tourism and tourists travel pattern and sentiment. Jinyan has been also working as research assistant on different projects related to social media and tourists' behavior.

Dr Susanne Becken is the Director of the Griffith Institute for Tourism and a Professor of Sustainable Tourism at Griffith University, Australia. Susanne has published widely in the field of sustainable tourism, is on the editorial board of 10 journals, and works closely with Government, businesses and international organisations on issues related to tourism management.

Bela Stantic is Professor in Computer Science and Director of "Big Data and Smart Analytics" Lab within the Institute of Integrated and Intelligent Systems at Griffith University. Bela is internationally recognized in field of Big Data analytics and efficient management of complex data structures. He successfully applied his research interdisciplinary to many areas including health, environment and tourism.

Received: October 15, 2018; Accepted: April 15, 2019.

Computer Science and Information Systems 16(2):657–688 https://doi.org/10.2298/CSIS181010014B

Automated Two-phase Business Model-driven Synthesis of Conceptual Database Models*

Drazen Brdjanin, Danijela Banjac, Goran Banjac, and Slavko Maric

University of Banja Luka, Faculty of Electrical Engineering
Patre 5, 78 000 Banja Luka, Republic of Srpska, Bosnia and Herzegovina
{drazen.brdjanin, danijela.banjac, goran.banjac, slavko.maric}@etf.unibl.org

Abstract. Existing approaches to business process model-driven synthesis of data models are characterized by a direct synthesis of a target model based on source models represented by concrete notations, where the synthesis is supported by monolithic (semi)automatic transformation programs. This article presents an approach to automated two-phase business process model-driven synthesis of conceptual database models. It is based on the introduction of a domain specific language (DSL) as an intermediate layer between different source notations and the target notation, which splits the synthesis into two phases: (i) automatic extraction of specific concepts from the source model and their DSL-based representation, and (ii) automated generation of the target model based on the DSL-based representation of the extracted concepts. The proposed approach enables development of modular transformation tools for automatic synthesis of the target model based on business process models represented by different concrete notations. In this article we present an online generator, which implements the proposed approach. The generator is implemented as a web-based, service-oriented tool, which enables automatic generation of the initial conceptual database model represented by the UML class diagram, based on business models represented by two concrete notations.

Keywords: BPMN, business process model, conceptual database model, domain specific language, extractor, generator, model-driven, service-oriented, UML.

1. Introduction

Data models are essential to any information system. The process of data modeling is not straightforward. It is often time-consuming and requires many iterations before the final model is obtained. Therefore, automatic data model design is very appealing and has been the subject of research for many years.

The majority of existing approaches to automated data model design are linguistics-based, since natural languages are commonly used for requirements specifications. However, their utilization is questionable for languages with complex morphology. Currently, there are several alternative approaches taking collections of forms or models (graphically specified requirements) as the basis for automated data model design, instead of textual specifications.

* This article constitutes an extended version of the conference paper entitled "*An Online Business Process Model-driven Generator of the Conceptual Database Model*" presented at the *8th International Conference on Web Intelligence, Mining and Semantics – WIMS'18*, June 25-27, 2018, Novi Sad, Serbia.

The idea of model-driven design of data models is already thirty years old [15]. However, the fully automatic *model-driven synthesis of the data model* (MDSDM) is still the subject of extensive research. In existing literature there is only a small number of papers presenting the implemented automatic model-driven generator of the target data model with corresponding evaluation results, while the great majority only present modest achievements in (semi)automated, or even manual, data model synthesis.

The existing MDSDM approaches are characterized by the direct data model synthesis based on source models represented by some concrete notation such as BPMN[1] or UML[2] activity diagram, where the synthesis is supported by monolithic transformation programs. Direct synthesis introduces dependency of the generation process from the source notation, because different source notations require different generators. Therefore, these generators depend on changes of notations, which are caused by metamodel changes and/or vendor specific implementations. The existing tools are also platform-dependent since they are mainly implemented as transformation programs deployed in some development platform (mainly Eclipse-based). Overcoming these shortcomings motivates our research with the following objectives: (1) define an approach to business process model-driven synthesis of conceptual database models, which enables the development of modular transformation tools for automatic synthesis of the target model based on source models represented by different notations with reduced dependency of the generation process from the source notations; (2) implement an online generator of conceptual database models based on the proposed approach, which facilitates database design as well as development of web-based environments for automatic business process model-driven database design.

In this article we present an approach that fulfills the aforementioned research objectives. It is based on the introduction of a *domain specific language* (DSL) as an intermediate layer between different source notations representing *business process models* (BPMs) and the target notation representing the *conceptual database model* (CDM). This simple DSL, called *Business Model Representation Language* (BMRL), is used to represent BPM concepts (such as participants, objects, tasks, etc.) having semantic potential for automatic CDM synthesis. With the introduction of BMRL, the CDM synthesis is split into two phases. In the first phase, specific concepts are to be extracted from the source BPM and represented by BMRL. Such an extraction can be easily implemented for different source notations. In this article we completely present extraction from BPMN models. In the second phase, the target CDM is to be generated based on the BMRL-based representation of the extracted BPM concepts. This generator is to implement a rather complex set of rules, which are dependent on simple and unique BMRL concepts, but independent of different source notations. This constitutes the first main contribution of the article.

The second main contribution of the article is related to the development and presentation of an online, publicly available, service-oriented CDM generator, which implements the proposed two-phase approach to BPM-driven CDM synthesis. Its usage could be twofold: (i) developers are able to implement their own applications consuming the exposed web service, (ii) database designers are able to use the implemented client application aimed at BPM-driven CDM synthesis.

---

[1] Business Process Model and Notation [50]

[2] Unified Modeling Language [51]

This article constitutes an extended and complete version of two closely related conference papers. The first paper [17], which proposed the first ideas about two-phase data model synthesis, was presented at the *7th International Conference on Model and Data Engineering – MEDI'17*. The second paper [18], which presented implementation of the first online two-phase BPM-driven CDM generator, was presented at the *8th International Conference on Web Intelligence, Mining and Semantics – WIMS'18*. The content of both conference papers is merged and expanded by: (1) a detailed presentation of the related work, (2) detailed presentation of the proposed approach with particular focus on BPMN as the starting notation, (3) recent achievements related to the implementation of the online generator applying the proposed approach, and (4) evaluation of the approach and implemented online generator.

The article is structured as follows. After the introduction, the second section presents the related work. The principles of the two-phase BPM-driven CDM synthesis are considered in the third section. The fourth section presents the semantic capacity of BPMs for automatic CDM synthesis, as a basis for DSL specification. The first phase of the CDM synthesis is presented in the fifth section, while the second phase is presented in the sixth section. The seventh section presents the implementation of the online generator. An illustrative example of the two-phase BPM-driven CDM synthesis is presented in the eighth section. The proposed approach is evaluated in the ninth section. Finally, the last section concludes the article.

2. Related Work

The survey [15] shows that existing MDSDM approaches, with respect to the primary focus of the source notation, can be classified as: *function-oriented*, *process-oriented*, *communication-oriented* and *goal-oriented*. The chronological overview of the existing MDSDM approaches, grouped by the source notation, is given in Fig. 1. Different marks are used to differentiate the source model completeness, which can be *complete* or *partial* (partial source model contains a single diagram, although a real model contains a finite set of diagrams). The figure also shows the level of automatisation for the approaches, which can be *manual* (not supported by any software tool), *semiautomatic* (supported by a tool, but designer's assistance is still required), or *automatic* (without designer's assistance). The arrows are used to emphasise the related papers presenting the improvements in the same approach.

Our approach belongs to the process-oriented approaches. As shown in Fig. 1, *process-oriented models* (POMs) constitute the largest category of models used as a source for MDSDM. Although the first data model synthesis based on a POM (A-graph) was proposed by Wrycza [64] in 1990, the boom of these approaches was influenced by the development of metamodel-based notations, particularly UML AD (Activity Diagram) and BPMN, as well as model-to-model transformation languages ATL[3] and QVT[4].

The survey [15] shows that POMs, used as a basis for data model synthesis, have been represented by seven different notations: BPMN, UML AD, Petri Net, RAD (Role Activity Diagram), GRAPES-BM/TCD, EPC (Event-driven Process Chain) and A-graph. Although there are more than 40 papers considering the POM-based MDSDM, the

[3] ATLAS Transformation Language [38]
[4] Query/View/Transformation [49]

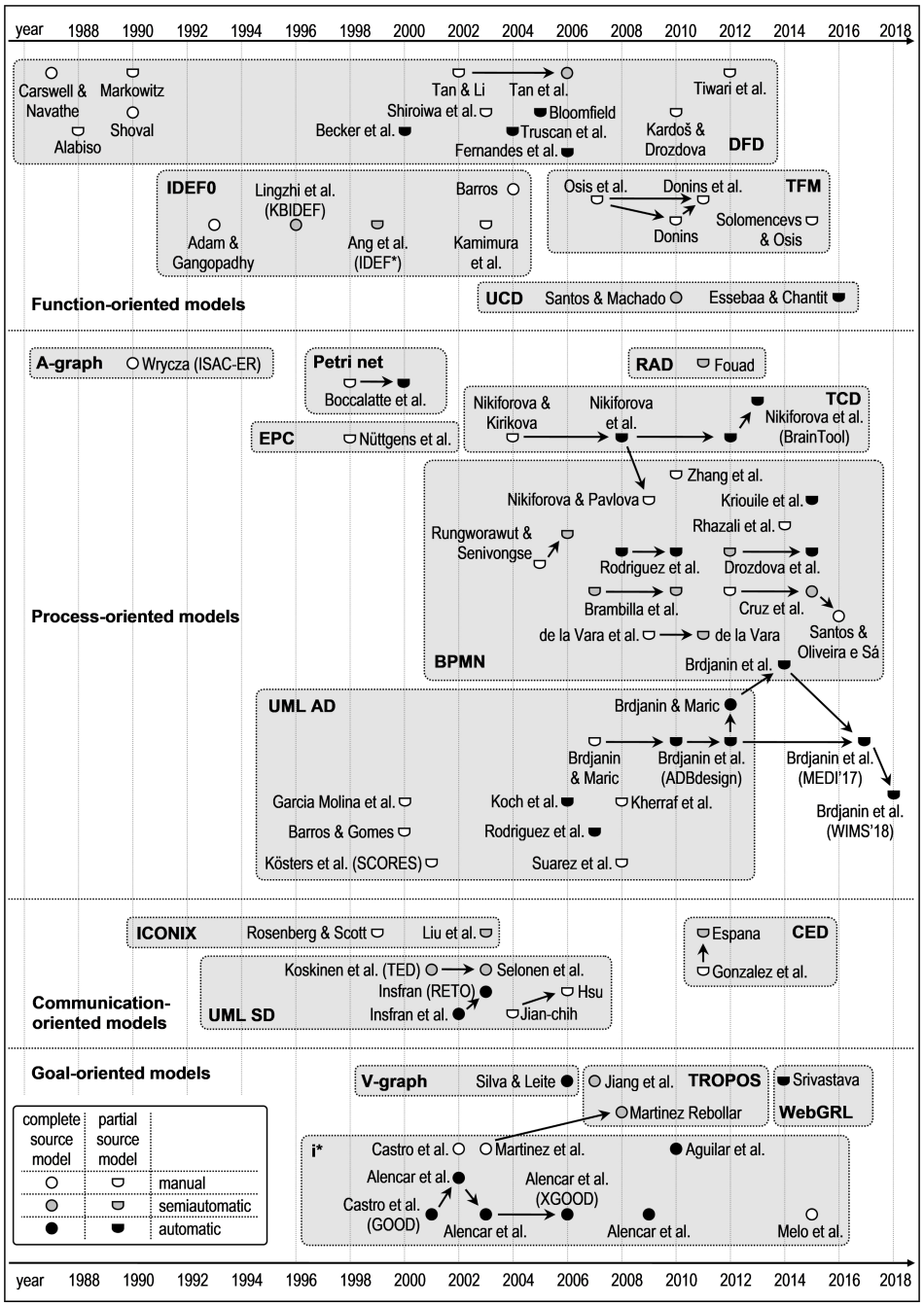


Fig. 1. Overview of existing MDSDM approaches

survey shows that the semantic capacity of POMs has still not been sufficiently identified to enable automatic synthesis of a complete data model. Furthermore, the majority of the approaches enable semiautomatic generation of the target model with modest completeness and precision. The majority of all POM-based approaches are also based on guidelines and informal rules that do not enable automatic MDSDM.

The BPMN is the most commonly used source notation for POM-based MDSDM. Among almost 20 papers, there are three QVT-based proposals [53, 54, 42], but with modest achievements in the automatic generation of analysis level class diagrams. There is an XSLT-based proposal [29] for automatic generation, as well as several proposals [55, 11, 12, 28, 30] for semiautomatic generation of class diagrams. A MDSDM based on BPMN is also considered in [65, 48, 30], but without implementation. The formal rules for automatic CDM synthesis based on BPMN are presented in [21, 56], and partially in [26, 27]. Other papers consider only the guidelines that do not enable automatic synthesis. A set of interrelated BPMs is considered in [27, 56], but the approaches are not implemented.

Among more than ten papers using UML AD as the source notation, only [23] presents an automatic CDM generator (named ADBdesign) based on the complete source model. Several papers [39, 40, 52, 16, 13, 22, 14] present the automatic, mainly ATL- and QVT-based, data model generation based on the incomplete source model, but with modest completeness and precision, while the others present only manual data model derivation.

There are also several related papers proposing the usage of TCD notation as a starting point for MDSDM, initially through an intermediate model, while [46] presents the BrainTool generator, which generates the data model directly from the TCD. However, like the majority, they do not consider the complete source model. Among the other POM-based approaches, there are only two papers [10, 34] presenting software tools for the (semi)automatic data model generation based on the partial source model.

The survey [15] shows that *function-oriented models* (FOMs), used as a basis for data model synthesis, have been represented by four different notations: DFD (Data Flow Diagram), SADT/IDEF0, TFM (Topological Functioning Model), and UML UCD (Use Case Diagram). Although the first ideas about the FOM-based MDSDM appeared in the second half of the 1980s, the survey shows that the semantic capacity of FOMs has not been sufficiently identified to enable automatic synthesis of the complete target data model. The large majority of the approaches are based on guidelines and informal rules and take an incomplete source model as the basis for data model synthesis. The automatic data model generation is presented in [8, 62, 33, 9, 32], while the semiautomatic generation is presented in [61, 43, 6, 57].

The survey [15] shows that *goal-oriented models* (GOMs), used as a starting point for data model synthesis, have been represented by the i* notation and some i*-originated notations like TROPOS, V-graph, and WebGRL. The automatic data model synthesis (to some extent) is presented in [25, 4, 3, 5, 59, 37, 45, 2, 1, 60]. The GOM-based approaches use complete source models.

The smallest number of models used for MDSDM are *communication-oriented models* (COMs). They have been represented by three different notations: UML SD (Sequence Diagram), CED (Communicative Event Diagram) and ICONIX (Robustness Diagram). UML SD is used in the majority of COM-based MDSDM approaches. The automatic data model synthesis is presented in [35, 36], while the semiautomatic synthesis is presented in [41, 58, 44, 31].

The large majority of all proposed MDSDM approaches are not evaluated at all (more than 90% according to [15]). Most of the papers reporting evaluation results mainly focus on approach usability, but not on the qualitative/quantitative assessment of the implemented tools or generated data models. The GOM-based approaches are not evaluated. Only one COM-based approach [31] is evaluated based on lab demos and a controlled experiment with master students (model completeness is $\sim$70%). Regarding the evaluation of the FOM-based approaches, only [61] presents evaluation results based on a controlled experiment (but the authors do not focus on the assessment of method effectiveness and efficiency). Regarding the evaluation of the POM-based approaches, three papers [21, 23, 47] report case-study based evaluation, while the results of controlled experiments are reported in [28, 34, 7, 19]. The most complete evaluation results of an MDSDM approach, which are based on the experiment conducted with a significant number of database practitioners, are presented in [19, 20] (average model completeness and precision are over 80%).

This article presents the recent achievements of an ongoing long-term research project about automatic BPM-driven CDM synthesis. The first ideas and prototype implementation (ADBdesign) were presented in 2010 [16]. The initial implementation was based on BPMs represented by UML ADs. The initial set of rules was upgraded, amended, and formalised in [14]. The automatic synthesis based on the finite set of UML ADs, was presented in [23]. The set of formal rules [14] was amended and applied [21] to collaborative BPMs represented by BPMN, and subsequently improved after a controlled experiment conducted with undergraduate students [7]. Through the experiment with students we obtained a very high completeness and precision of automatically generated CDMs (both average measures over 85%). After that experiment, we conducted the experiment [19, 20] with database practitioners, which almost confirmed the previous results. Based on the semantic capacity of POMs, which was identified and proved in the previous research and conducted experiments, we specified the aforementioned DSL named BMRL and proposed the two-phase BPM-driven approach to CDM synthesis [17]. This approach is depicted in Fig. 1 as notation-independent and outside of any POM region. In this article, we provide a detailed presentation of the approach, and the most recent implementation of the online CDM generator [18] based on the proposed approach.

In comparison to the existing approaches, the proposed approach is characterized by two-phase synthesis of the target model, while the existing approaches are characterized by direct synthesis. The implemented tool, in comparison to the existing tools, is the first online, web-based, publicly available tool. It is not dependent on any particular modeling platform, and enables automatic synthesis based on two different concrete source notations, in contrast to the existing tools, which are platform-dependent and enable (semi)automatic synthesis on the basis of a single starting notation. Furthermore, the implemented functionality is also available through the publicly available web service, which could be consumed from other modeling tools and platforms. Since the proposed two-phase approach is based on the previously experimentally evaluated approach to direct synthesis, this online tool is also characterized by very high effectiveness and efficiency, while the existing tools are not experimentally evaluated.

3. Two-phase BPM-driven CDM Synthesis

The MDSDM process is driven by a set of transformation rules combining two related sets of actions aimed at *extracting* characteristic concepts from the source model(s) and *generating* the corresponding concepts in the target model. In the existing MDSDM approaches, these two sets of actions are strongly coupled, meaning that the synthesis is performed by a single transformation program extracting concepts from the source model(s) and generating the target model. In the case of BPM-driven approaches, this means that a transformation program takes source BPMs represented by some concrete notation such as BPMN or UML AD, and generates the target data model represented by another concrete notation such as UML class diagram. Such *direct synthesis* has certain advantages such as directness and implementation facilitated by standardized transformation languages (e.g. ATL and QVT). However, it also requires different generators for different source notations. Therefore, these generators depend on modifications of the source and target notations (caused by metamodel changes and/or vendor specific implementations), and transformation rules as well. While the source notation-related modifications require modifications of only the corresponding generator, any modification of the transformation rules and/or target notation requires modifications of all generators, which can be considered a disadvantage of the direct synthesis.

In this article we argue that the above-described disadvantage can be reduced by decoupling the extracting and generating actions by separating them into two independent and consecutive activities, and splitting the synthesis into two phases. This can be achieved by introducing an intermediate layer between the source and target models. In the first phase, specific concepts are to be extracted from the source model(s) and represented at the intermediate layer. This can be achieved by transformation programs called *extractors*. A different extractor is required for each source notation. In the second phase, the target data model is to be generated based on the intermediate representation of the extracted concepts, which can be achieved by a single transformation program called *generator*. If we assume that the introduced intermediate layer is invariable[5], then the extractors depend only on the source notation-based modifications, while the generator depends on the modifications of the transformation rules and/or target notation. Like in the direct synthesis, the source notation-related modifications require modifications of only the corresponding extractor. However, any modification of the transformation rules and/or target notation requires modifications of only one generator. In this way the *indirect two-phase data model synthesis* can significantly reduce implementation efforts required to support diversity and/or modifications of the source notations and transformation rules for data model synthesis. Apart from the easier maintenance, the indirect synthesis could also be more effective in solving portability and interoperability issues, model checking, etc.

Alternatively to the proposed approach, some disadvantages of the direct synthesis could be reduced by applying a *hybrid approach*, which means that we may choose one (*primary*) source notation and implement the appropriate generator applying the principles of direct synthesis. The source models represented by other (*secondary*) modeling notations should be firstly transformed into the corresponding (equivalent)

[5] Strictly speaking, the intermediate layer is not immutable. Its changes occur with the additionally identified semantic capacity of BPMs for automatic CDM synthesis. However, these changes happen quite seldom compared with the related changes of source BPM notations.

models represented by the primary notation, and then the generator could be applied in order to obtain the target data model. For example, if we choose BPMN as the primary notation, then we need only the BPMN-based CDM generator, while the source models represented by some other notation (e.g. UML AD) should be firstly transformed into the corresponding BPMN models and subsequently used as the source for BPMN-based CDM synthesis. However, such a hybrid approach will also share the majority of the direct synthesis' disadvantages (*transformers* are dependent on both primary and secondary notations).

Following the idea of indirect data model synthesis, this article presents an approach to automated two-phase BPM-driven CDM synthesis. Although the intermediate layer could be differently represented, we use a DSL called *Business Model Representation Language* (BMRL) for its representation. It is a simple DSL for the representation of BPM concepts enabling the automatic CDM synthesis. Its specification is based on the results of our previous research [14, 21] indicating that BPMs are characterised by some typical *concepts* (such as *participants* and *objects*) and *facts* (such as *creation of objects* and *usage of objects*) that enable automatic CDM synthesis. Those concepts and facts are inherent to BPMs, but their representation may differ in different modelling languages. Independently of the used modelling notation, those concepts and facts have certain properties, meanings and roles (we use the *semantic capacity* term) that allow us to derive conclusions about the corresponding data model concepts (entity types, relationship types, etc.) and rules for mapping source BPMs to the target CDM.

With the introduction of BMRL, the CDM synthesis is split into two phases (Fig. 2). In the first phase, specific concepts are to be extracted from the source BPM and represented by BMRL. We illustrate the approach for two different notations (UML AD and BPMN) and provide details about the implemented extractors. In the second phase, the CDM (represented by the UML class diagram) is to be generated based on the BMRL-based representation of the extracted concepts. The generator has to implement a rather complex set of rules, which are dependent on simple and unique BMRL concepts, but independent of different source BPM notations.

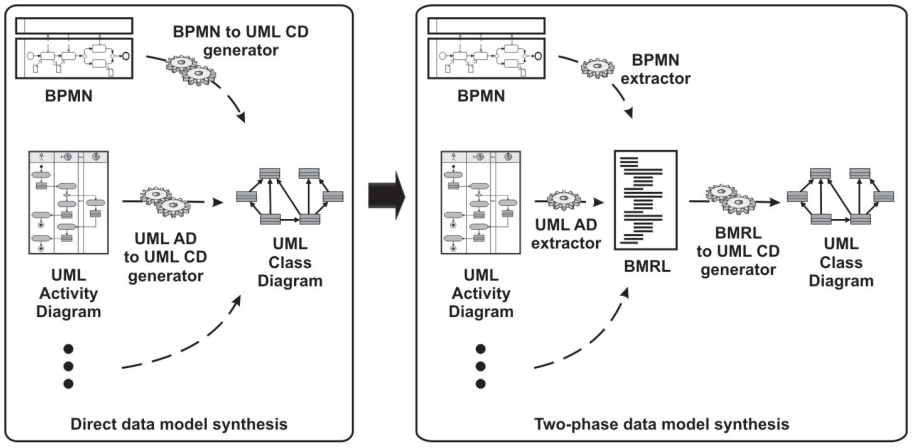


Fig. 2. Transition from direct (one-phase) to indirect (two-phase) data model synthesis

From the technical perspective (Fig. 3), the implemented extractors extract specific concepts from the source models conforming to the corresponding metamodels, and generate the corresponding BMRL representation of the extracted concepts according to the BMRL metamodel. The CDM generator generates the target model represented by UML class diagram conforming to the corresponding metamodel. As illustrated in Fig. 3, the proposed two-phase approach enables simple extensibility and support for other process-oriented notations (which are not necessarily metamodel-based) by implementing additional extractors.

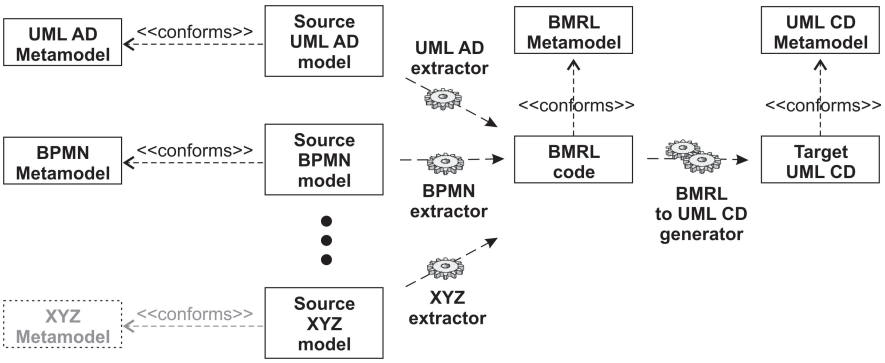


Fig. 3. Technical perspective of two-phase BPM-driven CDM synthesis

4. Semantic Capacity of BPMs for Automatic CDM Synthesis

As already stated, the previous research [14, 21, 7, 19, 20] implies that several common BPM concepts have semantic capacity for automatic CDM synthesis. This section provides a brief overview[6] and illustration (Fig. 4) of the identified semantic capacity of BPMs for automatic MDSDM. The identified semantic capacity constitutes the basis for the specification of BMRL and corresponding rules for both phases.

Typical BPM concepts that enable automatic generation of *entity types* (*classes*) in the target CDM are: *participants*, *roles*, *objects*, *message flows*, and *activations of existing objects*. Participants (may) have different roles. Participants and their roles are represented differently in BPMs (pools/lanes, partitions/subpartitions). All types of participants, and all their roles as well, are to be mapped into the corresponding classes in the target CDM (rule T_1). During the execution of a business process, participants perform tasks (actions) and exchange messages. Each different type of objects, and message flows as well, is to be mapped into the corresponding class in the target CDM (T_2). Each task/action may have a number of input and output objects that can be in different states. The objects can be *generated* in the given process, or *existing* – created in some other process. An activation represents the fact that an existing object constitutes input in a task that changes its state. Activated objects have the semantics similar to that of generated objects and need to be represented with a corresponding class (activation class) (T_3).

[6] A complete formal specification of transformation rules for direct BPM-driven CDM synthesis is given in [14, 21, 20].

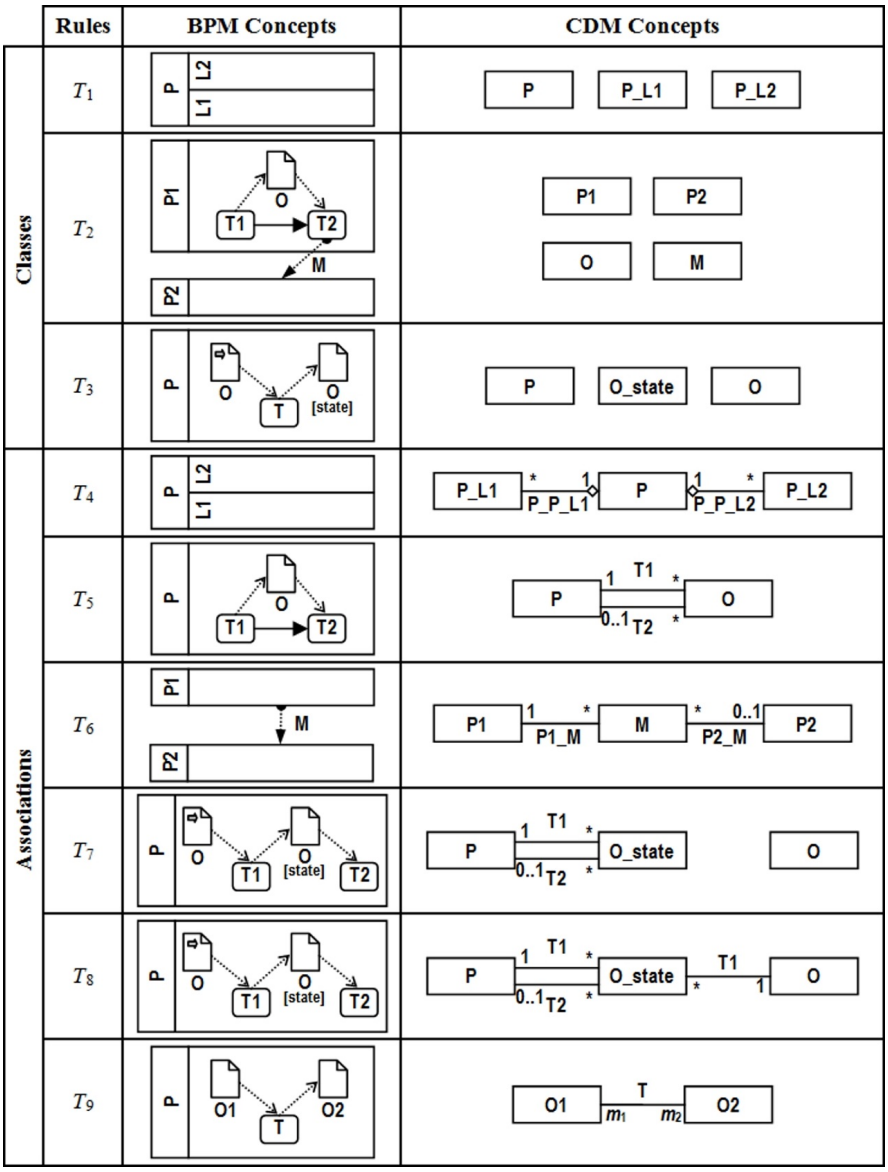


Fig. 4. Mapping of BPM concepts into CDM concepts

There are several common patterns in BPMs enabling automatic generation of *relationship types* (*associations*) in the target CDM. They enable generation of three types of associations: *participant-participant*, *participant-object*, and *object-object*. *Participant-participant* associations originate from the fact that a participant may have different roles. This implies that the class representing a pool should have associations with classes representing corresponding lanes (T_4). Process patterns having semantic potential for the generation of *participant-object* associations are: creation and subsequent usage of

generated objects (T_5), exchange of messages (T_6), and activation and subsequent usage of activated objects (T_7). Every mentioned fact is to be represented by corresponding association(s) with multiplicities 1:* or 0..1:*. There are two bases for the generation of *object-object* associations: (i) activation (T_8), which is represented with an association between the class that represents the existing object and the class that represents its activation, and (ii) tasks having input and output objects of different types (T_9), where the association end multiplicities depend on the nature of the objects (if they are either generated, non-activated existing or activated existing objects).

5. Phase I: DSL-based Representation of BPM Concepts

The first phase of the CDM synthesis includes the extraction of concepts from the source BPM and their BMRL-based representation. This section presents the BMRL specification and implementation of BPM extractors.

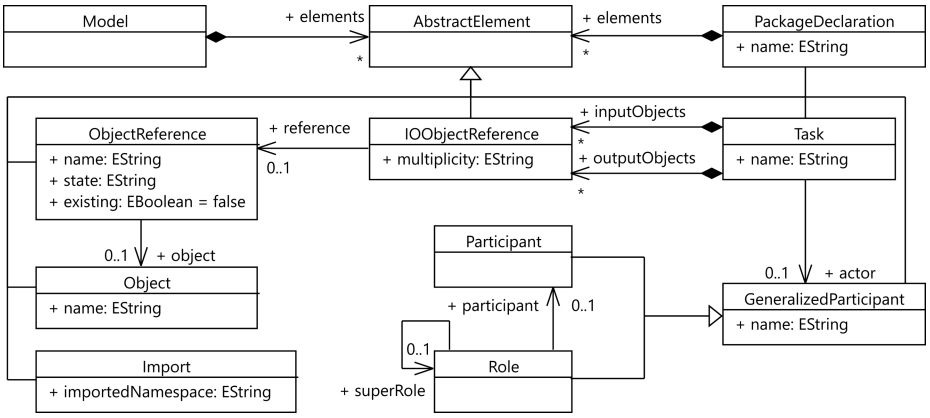
5.1. BMRL Specification

DSL is a computer programming language of limited expressiveness focusing on a particular domain. Programming languages, including DSLs, consist of three main elements: concrete syntax, abstract syntax and semantics [63]. The concrete syntax defines the notation with which users can express programs (it may be textual, graphical, tabular or combined). The abstract syntax is a data structure that can hold the semantically relevant information expressed by a program (most often represented by a tree or graph). There are two kinds of semantics. The static semantics is defined by a set of constraints and/or type system rules to which programs have to conform, while execution semantics refers to the meaning of a program once it is executed [63].

Based on the identified semantic capacity of BPMs for MDSDM, we defined a DSL named *Business Model Representation Language* (BMRL). For its specification we used Xtext[7] framework. Xtext belongs to parser-based approaches, in which a grammar specifies the sequence of tokens that forms structurally valid programs. In such systems, users interact only with concrete syntax, while the *abstract syntax tree* (AST) is constructed from the concrete syntax of a program [63]. Xtext relies on *Eclipse Modeling Framework* (EMF) [24] models for internal AST representation.

The Ecore metamodel of BMRL and its grammar are shown in Fig. 5. A BMRL program contains an arbitrary number of abstract elements: `PackageDeclaration`. `Import`, `GeneralizedParticipant`, `Task`, `Object`, `ObjectReference`, and `IOObjectReference`. Each business process participant can be represented by the `Participant` element. Participants can have roles (`Role`), and each role can have sub-roles. The `Participant` and `Role` elements have the `name` attribute. Participants (or participants with specified roles) perform tasks (`Task`). The `Task` element has the `name` attribute. Each task can have inputs and outputs, which are represented by input/output specification elements (`IOObjectReference`). Each different type of objects is represented by the `Object` element. For each type of objects, one or more references (`ObjectReference`) can be specified, since objects can be in different states (`state`). The `existing` attribute shows whether the given reference represents a reference to an

[7] http://www.eclipse.org/Xtext/




```
grammar org.unibl.etf.BMRL with org.eclipse.xtext.common.Terminals
generate bMRL "http://www.etf.unibl.org/bmrl2cd/BMRL"
Model:
    (elements+=AbstractElement)*;
PackageDeclaration:
    'package' name=QualifiedName '{' (elements+=AbstractElement)* '}';
AbstractElement:
    PackageDeclaration | Import | GeneralizedParticipant | Object |
    ObjectReference | Task;
QualifiedName:
    ID ('.' ID)*;
Import:
    'import' importedNamespace=QualifiedNameWithWildcard;
QualifiedNameWithWildcard:
    QualifiedName '.*'?;
GeneralizedParticipant:
    Participant | Role;
Participant:
    'participant' name=ID;
Role:
    'role' name=ID ('(' superRole=[Role|QualifiedName] ')' | 'of'
           participant=[Participant|QualifiedName]);
Object:
    'object' name=ID;
ObjectReference:
    'objectReference' name=ID 'references' object=[Object|QualifiedName]
                      ('[' state=ID ']')? (existing?='existing')?;
IOObjectReference:
    reference=[ObjectReference|QualifiedName]
              'multiplicity' multiplicity=Multiplicity;
Task:
    'task' name=ID '{'
    'actor' ':' actor=[GeneralizedParticipant|QualifiedName]
    ('input' '(inputObjects+=IOObjectReference)* '}')?
    ('output' '{' (outputObjects+=IOObjectReference)* '}')? '}';
Multiplicity:
    INT | '-1';
```

Fig. 5. BMRL metamodel (top) and grammar (bottom)

existing object, or to an object generated in the given BPM. The `IOObjectReference` references one of the `ObjectReference` elements and specifies its multiplicity. BMRL supports the use of packages (`PackageDeclaration`) and imports (`Import`) in order to avoid name ambiguities.

5.2. BPM Extractors

As previously described, the first phase includes the extraction of important concepts from the source BPM and their representation according to the implemented DSL. This extraction and generation of the corresponding BMRL code can be implemented in different ways, either by using general purpose or specialized transformation languages. We used Acceleo[8] for the implementation of extractors. So far, we have implemented two extractors – one for BPMN and another one for UML AD. In this article we provide implementation details only for the BPMN extractor[9].

The BPMN extractor performs the extraction of characteristic concepts from the source BPM represented by BPMN (the related BPMN metamodel [50] excerpt is shown in Fig. 6), and generates the corresponding BMRL representation of the extracted concepts.

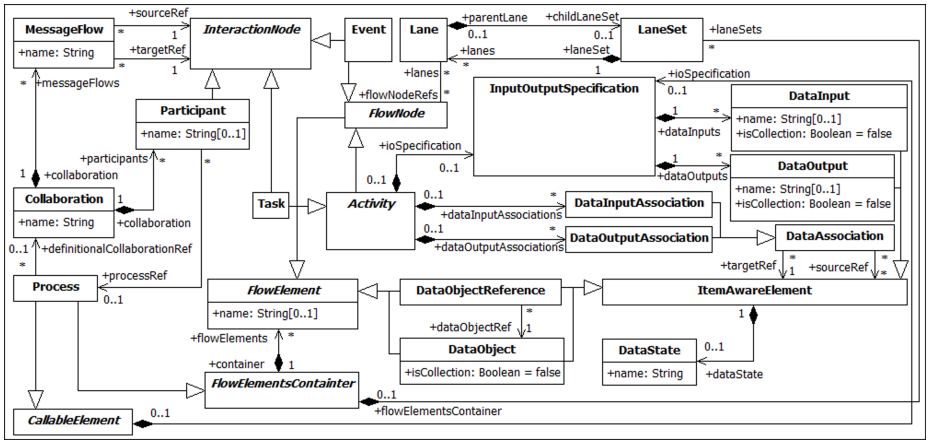


Fig. 6. BPMN metamodel [50]

The formal rules for the extraction of concepts from BPMN and their BMRL-based representation are given in Tables 1 and 2. This set of rules enables the extraction of: participants and their roles, generated and existing objects, as well as tasks having input and/or output objects.

For each `Participant` in the source model, the corresponding `Participant` element (of the same name) has to be generated in the BMRL. For each `Lane` the corresponding `Role` element has to be generated. The generated `Role` belongs to the `Participant` that corresponds to the `Participant` containing the given `Lane`.

Each `Task` may have input and/or output objects (`DataInput`, `DataOutput`, `DataObjectReference`, and `MessageFlow`), which can be in different states. For each `DataInput`, `DataOutput`, `MessageFlow`, and `DataObject` referenced by `DataObjectReference`, the corresponding `Object` and `ObjectReference` elements have to be generated. The `Object` element represents different type of objects

[8] http://www.eclipse.org/acceleo/

[9] Implementation details for the UML AD extractor are presented in [17].

and it has to be named the same as the given source element. The `ObjectReference` element represents the concrete object and it should be named by concatenating the names of the object and its state. Attributes `state` and `existing` of the `ObjectReference` depend on the state and nature (generated/existing) of the given source element. If it represents the existing object, then its name should be prefixed with `'Existing'`.

For each `Task` element in the source model, the corresponding `Task` has to be generated, whose actor attribute corresponds to the `GeneralizedParticipant` performing the given task, while the `inputObjects` and `outputObjects` attributes are represented by the `IOObjectReference` elements generated for `dataInput-Associations` and `dataOutputAssociations` elements of the task, as well as for `MessageFlows` referencing the given task. The `multiplicity` attribute has to be set based on the `isCollection` attribute of the element referenced by the (input or output) data association or message flow.

For each `MessageFlow` element referencing `Participant` or `Event` in the source model, the corresponding `Task` (named `'SendMessage'` or `'ReceiveMessage'`) has to be generated. Actor of the generated task is `GeneralizedParticipant` which corresponds to the `Participant` referenced by the message flow, or the `Participant` related with the `Event` referenced by the message flow.

Figure 7 (on the left) provides an illustration of the extraction of specific concepts from a simple BPM (BPMN) and generation of the corresponding BMRL code. This simple BPM represents an activation of the existing object `Book`, which is performed by the `Librarian` participant. The dashed arrows depict the mapping of source BPMN concepts into the target BMRL concepts.

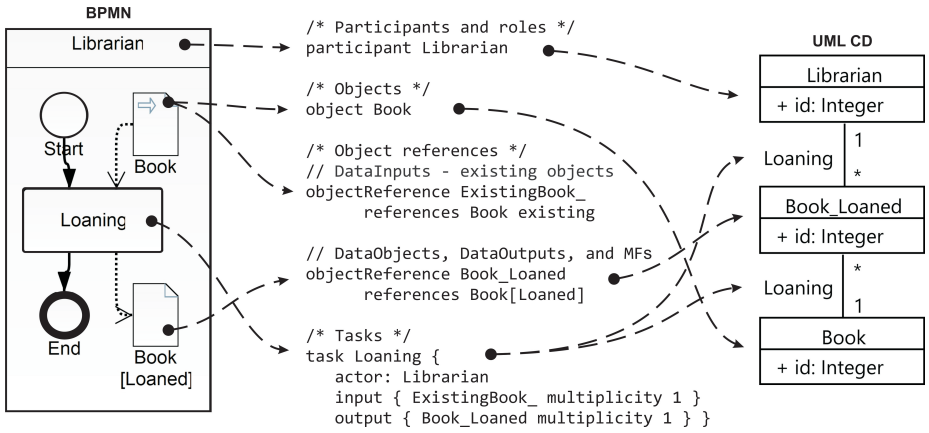


Fig. 7. From BPM (BPMN) through BMRL to CDM (UML CD)

Table 1. Rules for extraction of concepts from BPMN and their BMRL representation

Source concept in BPMN	**Target BMRL concept**
`sp:Participant`	`p:Participant` `{p.name=sp.name}`

`l:Lane`
`{l∈sp.processRef.laneSets.lanes}`
`ls:Lane`
`{ls∈l.childLaneSet.lanes}`
	`r:Role`
	`{r.name=l.name ∧ r.participant=p}`
	`rs:Role`
	`{rs.name=ls.name ∧ rs.superRole=r}`

`di:DataInput`
`{di∈sp.processRef.ioSpecification.dataInputs}`
	`o:Object`
	`{o.name=di.name}`
	`or:ObjectReference`
	`{or.object=o ∧ or.existing=true ∧`
	`or.name=concat('Existing',di.name,di.dataState.name) ∧`
	`or.state=di.dataState.name}`

`do:DataOutput`
`{do∈sp.processRef.ioSpecification.dataOutputs}`
	`o:Object`
	`{o.name=do.name}`
	`or:ObjectReference`
	`{or.object=o ∧ or.existing=false ∧`
	`or.name=concat(do.name,do.dataState.name) ∧`
	`or.state=do.dataState.name}`

`mf:MessageFlow`	`o:Object`
	$\{\,\mathbf{o.name} = \begin{cases} \mathbf{mf.name}, \mathbf{isUndefined(mf.messageRef)} \\ \mathbf{mf.messageRef.name},\ \text{otherwise} \end{cases}\,\}$
	`or:ObjectReference`
	`{or.object=o ∧ or.existing=false ∧`
	$\mathbf{or.name} = \begin{cases} \mathbf{mf.name}, \mathbf{isUndefined(mf.messageRef)} \\ \mathbf{mf.messageRef.name},\ \text{otherwise} \end{cases}\,\}$

`dor:DataObjectReference`	`o:Object`
	`{o.name=dor.dataObjectRef.name}`
	`or:ObjectReference`
	`{or.object=o ∧ or.existing=false ∧`
	`or.name=concat(dor.dataObjectRef.name,dor.dataState.name) ∧`
	`or.state=dor.dataState.name}`

Table 2. Rules for extraction of concepts from BPMN and their BMRL representation (continued)

Source concept in BPMN	Target BMRL concept
st:Task, in:BaseElement, out:BaseElement {in ∈ st.dataInputAssociations.sourceRef ∪ {mf:MessageFlow | mf.targetRef=t} ∧ out ∈ st.dataOutputAssociations.targetRef ∪ {mf:MessageFlow | mf.sourceRef=t} }	t:Task {t.name=st.name ∧ t.actor=(p ∨ r ∨ rs) ∧ iior∈t.inputObjects ∧ iior.reference=or ∧ iior.multiplicity=im ∧ oior∈t.outputObjects ∧ oior.reference=or ∧ oior.multiplicity=om, $\texttt{im} = \begin{cases} *, \texttt{C1} \\ 1, \text{ otherwise} \end{cases}$, $\texttt{om} = \begin{cases} *, \texttt{C2} \\ 1, \text{ otherwise} \end{cases}$, C1=in.isCollection ∨ ¬isUndefined(in.dataObjectRef) ∧ in.dataObjectRef.isCollection ∨ ¬isUndefined(in.messageRef) ∧ ¬isUndefined(in.messageRef.itemRef) ∧ in.messageRef.itemRef.isCollection, C2=out.isCollection ∨ ¬isUndefined(out.dataObjectRef) ∧ out.dataObjectRef.isCollection ∨ ¬isUndefined(out.messageRef) ∧ ¬isUndefined(out.messageRef.itemRef) ∧ out.messageRef.itemRef.isCollection}
mf:MessageFlow {typeof(mf.sourceRef) ∈ {Partcipant,Event}}	t:Task {t.name=concat('SendMessage',S1) ∧ t.actor=(p ∨ r ∨ rs) ∧ oior∈t.outputObjects ∧ oior.reference=or ∧ oior.multiplicity=om, $\texttt{S1} = \begin{cases} \textbf{mf.name}, \textbf{isUndefined(mf.messageRef)} \\ \textbf{mf.messageRef.name}, \text{ otherwise} \end{cases}$, $\texttt{om} = \begin{cases} *, \texttt{C3} \\ 1, \text{ otherwise} \end{cases}$, C3=¬isUndefined(mf.messageRef) ∧ ¬isUndefined(mf.messageRef.itemRef) ∧ mf.messageRef.itemRef.isCollection }
mf:MessageFlow {typeof(mf.targetRef) ∈ {Partcipant,Event}}	t:Task {t.name=concat('ReceiveMessage',S2) ∧ t.actor=(p ∨ r ∨ rs) ∧ iior∈t.inputObjects ∧ iior.reference=or ∧ iior.multiplicity=im, $\texttt{S2} = \begin{cases} \textbf{mf.name}, \textbf{isUndefined(mf.messageRef)} \\ \textbf{mf.messageRef.name}, \text{ otherwise} \end{cases}$, $\texttt{im} = \begin{cases} *, \texttt{C4} \\ 1, \text{ otherwise} \end{cases}$, C4=¬isUndefined(mf.messageRef) ∧ ¬isUndefined(mf.messageRef.itemRef) ∧ mf.messageRef.itemRef.isCollection }

6. Phase II: DSL-based CDM Synthesis

The second phase includes automatic generation of the target CDM represented by UML class diagram (related UML metamodel [51] excerpt is shown in Fig. 8) based on the BMRL representation of the source BPM.

The rules for the automatic generation of the UML class diagram, based on the BMRL representation of the source BPM, are given in Tables 3 and 4. The first two columns contain source and target concepts, while the third column contains labels for the corresponding mapping rules illustrated in Fig. 4. An informal description of these rules is given in Sect. 4.

The formal rules constitute the basis for the implementation of the CDM generator. The generator can be implemented in different ways. In our case, it is implemented as an automatic Xtend-based[10] generator.

Figure 7 above (right side) provides an illustration of the automatic generation of the UML class diagram. The dashed arrows are used to illustrate mappings of BMRL concepts into the UML concepts.

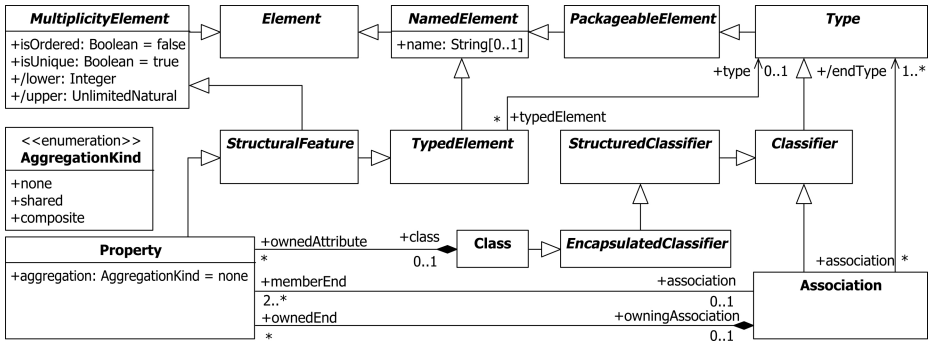


Fig. 8. UML CD metamodel [51]

Table 3. Mapping of BMRL concepts into UML class diagram concepts

Source BMRL concepts	Target UML CD concepts	Rules
`p:Participant`	`ep:Class` `{ep.name=p.name}`	T_1
`r:Role` `{r.participant=p}`	`er:Class` `{er.name=concat(r.participant.name,r.name)}`	T_1

	`rpr:Association`
	`{rpr.name=concat(r.participant.name,er.name) ∧`
	`rpr.memberEnd.source=ep ∧`
	`multiplicity(rpr.memberEnd.source)=1 ∧`
	`rpr.memberEnd.target=er ∧`
	`multiplicity(rpr.memberEnd.target)=*}`
		T_4

[10] http://www.eclipse.org/xtend/

Table 4. Mapping of BMRL concepts into UML class diagram concepts (continued)

Source BMRL concepts	Target UML CD concepts	Rules
o:Object	eo:Class {eo.name=o.name}	T_2
t:Task, in:IOObjectReference[0..*], out:IOObjectReference[0..*] {in=t.inputObjects ∧ out=t.outputObjects ∧ in.reference.existing=true ∧ in.reference.object= out.reference.object}	ea:Class {ea.name=concat(in.reference.object.name, out.reference.state)}	T_3
	rpa:Association {rpa.name=t.name) ∧ rpa.memberEnd.source=(ep ∨ er) ∧ multiplicity(rpa.memberEnd.source)=1 ∧ rpa.memberEnd.target=ea ∧ multiplicity(rpa.memberEnd.target)=∗}	T_7
	roa:Association {roa.name=t.name) ∧ roa.memberEnd.source=eo ∧ multiplicity(roa.memberEnd.source)=1 ∧ roa.memberEnd.target=ea ∧ multiplicity(roa.memberEnd.target)=∗}	T_8
t:Task, in:IOObjectReference[0..*], out:IOObjectReference[0..*] {in=t.inputObjects ∧ out=t.outputObjects ∧ ∄in ∣ in.reference.object= out.reference.object}	rgc:Association {rgc.name=t.name) ∧ rgc.memberEnd.source=(ep ∨ er) ∧ multiplicity(rgc.memberEnd.source)=1 ∧ rgc.memberEnd.target=eo ∧ multiplicity(rgc.memberEnd.target)=∗}	T_5 T_6
t:Task, in:IOObjectReference[0..*] {in=t.inputObjects ∧ in.reference.existing=false}	ru:Association {ru.name=t.name) ∧ ru.memberEnd.source=(ep ∨ er) ∧ multiplicity(ru.memberEnd.source)=0..1 ∧ ru.memberEnd.target=(eo ∨ ea) ∧ multiplicity(ru.memberEnd.target)=∗}	T_5 T_6 T_7
t:Task, in:IOObjectReference[0..*], out:IOObjectReference[0..*] {in=t.inputObjects ∧ out=t.outputObjects ∧ in.reference.object≠ out.reference.object}	roo:Association[n] {roo.name=t.name) ∧ roo.memberEnd.source=(eo ∨ ea) ∧ multiplicity(roo.memberEnd.source)=sm ∧ roo.memberEnd.target=(eo ∨ ea) ∧ multiplicity(roo.memberEnd.target)=tm} $\mathtt{n} = \begin{cases} 1, \ \mathtt{in.multiplicity} \in \{1, *\} \\ \mathtt{in.multiplicity}, \ \text{otherwise} \end{cases}$ $\mathtt{low(sm)} = \begin{cases} 0, \mathtt{in.multiplicity} = * \vee \\ \quad \exists \mathtt{r} \in \mathtt{in} \mid \mathtt{r.reference.object} = \\ \qquad \mathtt{out.reference.object} \wedge \\ \qquad \mathtt{r.reference.existing} = \mathtt{false} \\ 1, \ \text{otherwise} \end{cases}$ $\mathtt{high(sm)} = \begin{cases} *, \ \mathtt{in.multiplicity} = * \\ 1, \text{otherwise} \end{cases}$ $\mathtt{low(tm)} = 0$ $\mathtt{high(tm)} = \begin{cases} *, \ \mathtt{out.multiplicity} \neq 1 \vee \\ \quad \mathtt{in.reference.existing} = \mathtt{true} \\ 1, \text{otherwise} \end{cases}$	T_9

7. Online Two-phase BPM-driven CDM Generator

The proposed approach enables implementation of a modular tool for BPM-driven CDM synthesis, which consists of loosely coupled components aimed at automatic extraction of specific concepts from the source BPMs represented by different concrete notations, their BMRL-based representation, and automatic CDM generation.

Like all tools in the existing tool-supported MDSDM approaches, our initial set of tools, presented in [17], was also platform-dependent, since all tools were implemented as Eclipse plug-ins. In order to obtain a platform-independent and publicly available tool for the BPM-driven CDM synthesis, we performed the migration of these tools into a SOA[11] application.

7.1. Architecture of Online Generator

The online generator is implemented as an orchestration of web services. Its architecture is presented in Fig 9. We used the REST architectural style for implementation of services. In a positive scenario, the orchestrator service receives a source BPM represented by BPMN or UML AD (`input.bpmn`/`input.uml`), and returns the corresponding CDM (`cdm.uml`) to the caller.

In the first phase, the orchestrator service sends the source BPM to the corresponding extractor service, which takes the XMI representation of the source model, generates the corresponding BMRL code (`input.bmrl`) and returns it to the orchestrator service. Currently, two extractor services (shown as `BPMN extractor` and `UML AD extractor` in Fig 9) are implemented. Implementation is based on the Java archives (JAR) obtained by exporting the Acceleo-based extractors [17]. The proposed architecture enables easy extension of the online generator by additional extractors aimed at extraction of characteristic concepts from BPMs represented by other notations.

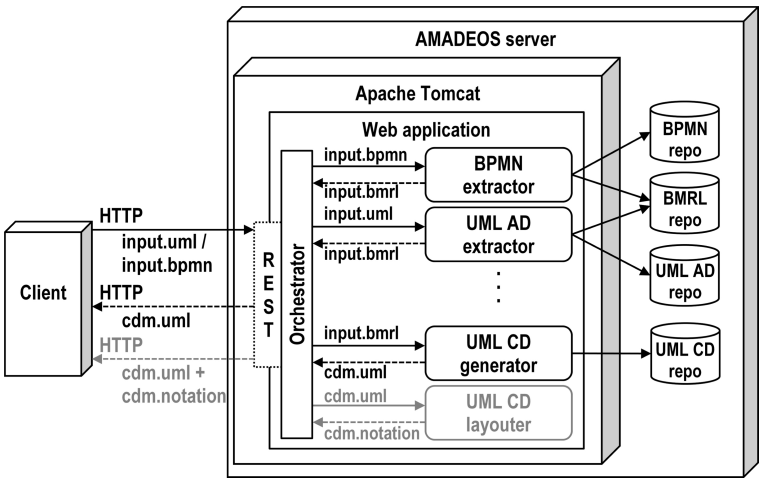


Fig. 9. Architecture of the online CDM generator

[11] Service-oriented architecture

In the second phase, the orchestrator service sends the generated BMRL representation (`input.bmrl`) to the generator service (`UML CD generator`), which takes the BMRL code, generates the XMI representation of the target CDM (`cdm.uml`) and returns it to the orchestrator service. Implementation of the generator service is based on the Java archive obtained by exporting the Xtend-based generator [17].

Each extractor stores the source model and generated BMRL code in appropriate server repositories. The generator service stores the generated CDM in the corresponding repository, as well. These repositories will be used in the future for further analysis of the approach and implemented system.

Currently, we are developing the online generator further. The next release will include the layouter service (`UML CD layouter` in Fig 9) aimed at automatic generation of the layout of the UML class diagram (`cdm.notation`), in order to enable visualization of the automatically generated CDM in the browser.

7.2. Usage of Online Generator

The usage of the implemented online generator can be twofold. In the first scenario, developers are able to implement their own applications invoking the exposed web service that orchestrates the two-phase CDM synthesis. In the second scenario, database designers are able to use the implemented client application.

For the first usage scenario, the online CDM generator[12] exposes one method for the target CDM generation, which accepts the `multipart/form-data media` type. The request should consist of two required named body parts `"source_model_type"` and `"input"`. The `"source_model_type"` body part defines the type of the input model. The permitted values are `"AD"` for UML AD and `"BPMN"` for BPMN. The `"input"` body part is the uploaded source model file. In the case of a successful generation of the target model, the service responds with status 200 (`OK`) and produces the `application/octet-stream` media type representing the generated target model. In the case of any error, the service responds with status 204 (`no content`)[13]. An example of the service client[14] is given in Fig 10.

The second usage scenario of the online generator is a client application[15] (Fig 11). Through this application database designers are able to upload the source BPM and download the XMI-representation of the automatically generated CDM, which can subsequently be imported and visualized in a certain modeling tool/platform. The visualization and editing functionalities of automatically generated models in the web browser are not currently supported by the implemented client application. The relevant work is underway.

---

[12] The implemented online generator is available at:
`http://m-lab.etf.unibl.org:8080/amadeos/services/generate/cdm`

[13] We would like to emphasize the fact that the currently supported BPM specifications are BPMN 2.0 [50] and UML 2.5 [51]. However, it is possible that the generator will return status 204 in cases of some vendor's specificities. Currently, we are developing robust extractor services in order to overcome problems related to platform and vendor serialization specificities.

[14] In order to facilitate development of client applications consuming the implemented online generator, as well its usage, some sample source models, Eclipse-based modeling platform and sample client code are available at GitLab: `https://gitlab.com/m-lab-research/amadeos`

[15] The client application is available at:
`http://m-lab.etf.unibl.org:8080/amadeos/generator.html`

```
FileDataBodyPart filePart =
       new FileDataBodyPart("input", new File("path_to_source_model"));
FormDataMultiPart multipart = new FormDataMultiPart();
multipart.field("source_model_type", "AD").bodyPart(filePart);
// For BPMN: multipart.field("source_model_type", "BPMN").bodyPart(filePart);

ClientConfig clientConfig = new ClientConfig().register(MultiPartFeature.class);
Client client = ClientBuilder.newClient(clientConfig);
String server = "http://m-lab.etf.unibl.org:8080/amadeos/services/";
WebTarget target = client.target(server).path("generate").path("cdm");

Response response =
    target.request().post(Entity.entity(multipart, multipart.getMediaType()));

if (response.getStatus() == 200) {
    InputStream is = response.readEntity(InputStream.class);
    File f = new File("path_to_target_model.uml");
    FileUtils.copyToFile(is, f);
    is.close();
}

filePart.cleanup();
multipart.close();
client.close();
response.close();
```

Fig. 10. An example of the service client

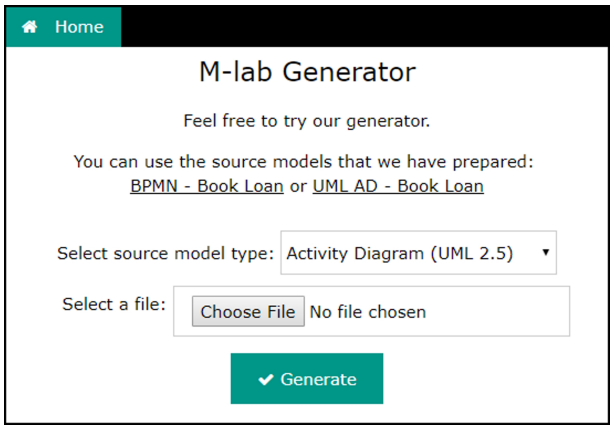
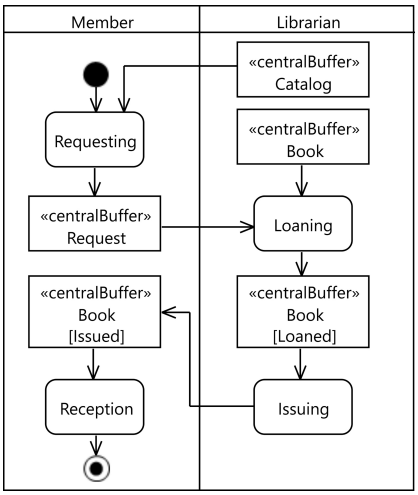


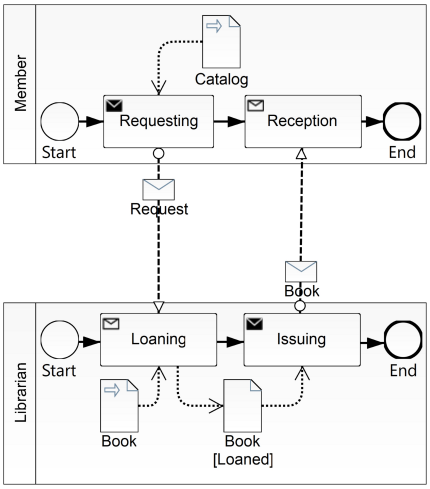
Fig. 11. Screenshot of the client application form

8. Illustrative Example of Two-phase BPM-driven CDM Synthesis

This section presents an illustrative example of the proposed approach. This example aims to illustrate the process of two-phase synthesis as such, and to prove that BPMs, regardless of the used notation, have the semantic capacity for automatic CDM synthesis. We prepared two simple BPMs represented by two concrete notations (UML AD and BPMN), which are currently supported by the online tool. Both BPMs represent the same business process (Book loaning)[16]. These two models are shown on the top of Fig. 12.

[16] A detailed description of these sample models is omitted due to their simplicity.





UML AD extractor

BPMN extractor

```
participant Librarian
participant Member

object Book
object Request
object Catalog

objectReference ExistingCatalog_
      references Catalog existing
objectReference ExistingBook_
      references Book existing
objectReference Request_
      references Request
objectReference Book_Issued
      references Book[Issued]
objectReference Book_Loaned
      references Book[Loaned]

task Loaning {
 actor: Librarian
 input { Request_ multiplicity 1
         ExistingBook_ multiplicity 1 }
 output { Book_Loaned multiplicity 1 } }
task Reception {
 actor: Member
 input { Book_Issued multiplicity 1 }
 output { } }
task Issuing {
 actor: Librarian
 input { Book_Loaned multiplicity 1 }
 output { Book_Issued multiplicity 1 } }
task Requesting {
 actor: Member
 input { ExistingCatalog_ multiplicity 1 }
 output { Request_ multiplicity 1 } }
```

```
participant Librarian
participant Member

object Book
object Request
object Catalog

objectReference ExistingBook_
      references Book existing
objectReference ExistingCatalog_
      references Catalog existing
objectReference Book_
      references Book
objectReference Request_
      references Request
objectReference Book_Loaned
      references Book[Loaned]

task Reception {
 actor: Member
 input { Book_ multiplicity 1 }
 output { } }
task Issuing {
 actor: Librarian
 input { Book_Loaned multiplicity 1 }
 output { Book_ multiplicity 1 } }
task Requesting {
 actor: Member
 input { ExistingCatalog_ multiplicity 1 }
 output { Request_ multiplicity 1 } }
task Loaning {
 actor: Librarian
 input { ExistingBook_ multiplicity 1
         Request_ multiplicity 1 }
 output { Book_Loaned multiplicity 1 } }
```

Fig. 12. BMRL representation of sample BPMs: UML AD (left) and BPMN (right)

Apart from the sample BPMs, Fig. 12 also shows their BMRL representation (bottom) automatically generated by the implemented extractors. In this way, Fig. 12 illustrates the first phase of the proposed approach.

Figure 13 depicts the result of the second phase of BPM-driven CDM synthesis. The depicted class diagram is visualized by the Papyrus[17] tool in the Eclipse IDE. It represents the automatically generated CDM based on the BMRL representation of the sample source BPM(s) shown in Fig. 12.

The fact that the same CDM represents the result of the application of the implemented tools to both sample BPMs (although represented by two different notations), proves the hypothesis that BPMs, regardless of the modeling notation, are characterized by some common concepts and facts having semantic capacity for automatic CDM synthesis.

The sample BPMs constitute the simplified models of the book loaning process. Consequently, the automatically generated CDM also constitutes a simplified version of the corresponding target CDM. Given the model simplicity, we do not provide a detailed analysis and evaluation of the automatically generated CDM, particularly its completeness. However, regardless of its simplicity, the correctness of the automatically generated CDM is very high.

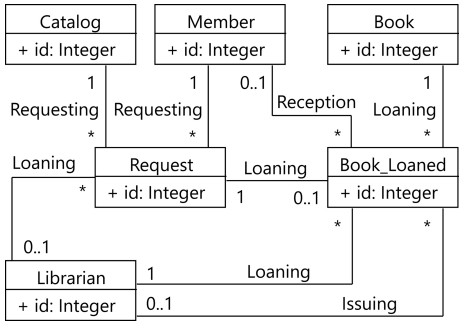


Fig. 13. UML class diagram representing the automatically generated CDM based on the source BPM(s) shown in Fig. 12

9. Verification and Validation

In this section we evaluate the proposed approach to automated two-phase BPM-driven CDM synthesis based on the experimental evaluation of the implemented online CDM generator.

9.1. Verification

Verification is the process of checking that the software meets the specification. We verified the implemented online two-phase generator against the existing direct ATL-based generator [20].

[17] https://eclipse.org/papyrus/

In order to verify implementation of the online generator, we applied it on real BPMs. Here we provide a real BPMN model (Fig. 14) of order processing, which was also used in the experiment conducted with database professionals [19, 20]. Although the presented workflow (Fig. 14) is quite intuitive, we still provide a short description. The given model represents an online purchasing and selling business process, with deferred payment option assumed. The process starts with the customer online order specification, which consists of a header and order details. After the order has been created, the salesman checks the customer's status (validity, creditworthiness, etc.) and availability of ordered items in stock. Based on the performed checks, the salesman decides whether the order is acceptable or not (in the latter case the order is canceled). If the order is acceptable, the invoice (consisting of a header and invoice details) is created and the stockman starts collecting and packing for shipment and delivery stock items for all confirmed order details. After all items have been prepared for delivery, the driver picks up the documentation and loads and delivers them to the customer. After delivery the customer confirms receipt of goods and the process finishes with setting the related document status to delivered.

Figure 15 depicts a class diagram (visualised by Papyrus) representing the automatically generated CDM based on the BMRL representation of the source BPMN model of order processing. This CDM, automatically generated by the implemented online two-phase generator, is equal to the CDM automatically generated by the direct ATL-based generator [20]. By applying the same verification procedure for other BPMs, we also obtain the complete matching of the corresponding generated CDMs. This fact, that we obtain equal CDMs by applying both two-phase generator and direct ATL-based generator, confirms that the online two-phase generator properly implements the same functionality of the automatic BPM-driven CDM synthesis.

9.2. Validation

Validation is the process of checking whether the specification captures the customer's needs. In the context of evaluation of the proposed approach and implemented online generator, validation could be twofold – from perspectives of two different classes of users: database designers and developers.

Validation from the database designers' perspective includes an assessment of the effectiveness and efficiency of the implemented online generator. Since both online CDM generator and the direct ATL-based generator [20] generate the same two CDMs based on the same BPM, the effectiveness of the two-phase BPM-driven synthesis is equal to the effectiveness of the direct BPM-driven synthesis. The efficiency is similar due to the equal complexity[18] of the approaches.

Here we refer to the main results of the experiment conducted with database practitioners [19, 20] in order to evaluate the direct BPM-driven CDM synthesis and direct ATL-based generator [20]. The evaluation was twofold. Firstly, it focused on the assessment of approach effectiveness, through the assessment of correctness (*precision*) and completeness (*recall*) of the automatically generated model. The average effectiveness (*F-measure*) was $\sim$78% for automatic generation of classes, and $\sim$85% for

[18] Both approaches have linear complexity ($O(n)$).

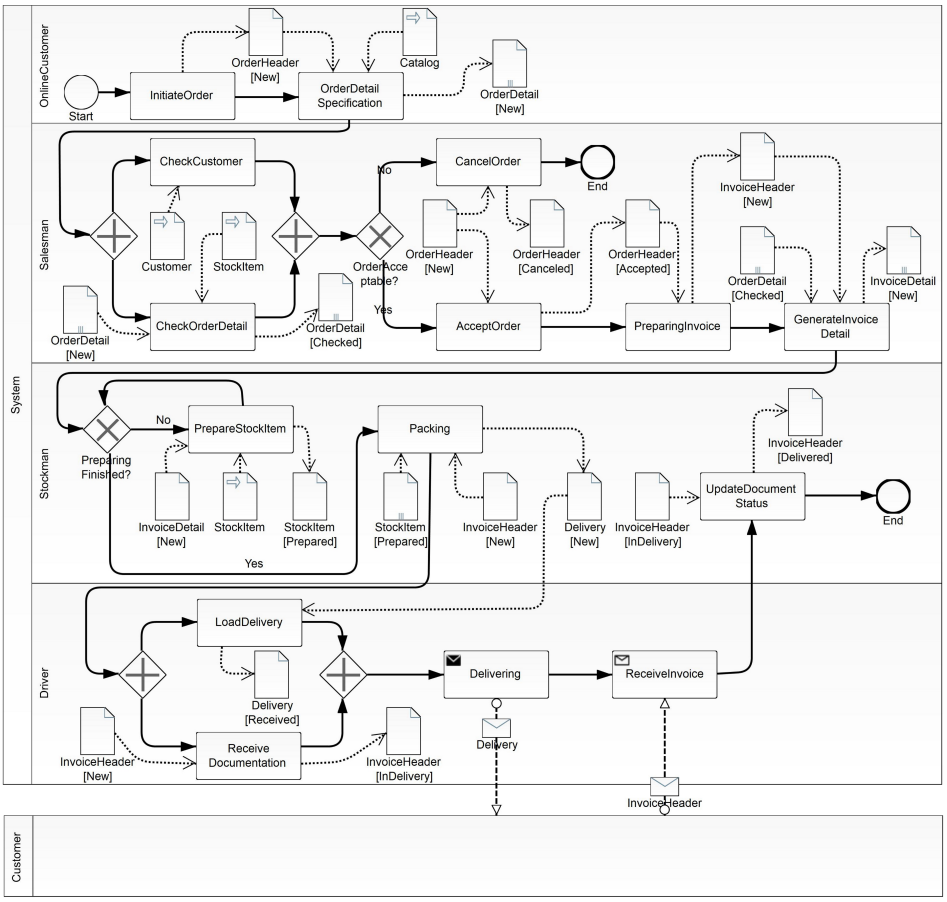


Fig. 14. BPMN model of order processing [19, 20]

associations. The average recall of the generated model was above 80%. The average precision for automatically generated classes was above 75%, while the average precision for associations was about 90%. Secondly, it focused on the assessment of usability of the automatically generated model as a starting base for manual design of the target model, as well as the assessment of efficiency of such an approach in contrast to the manual design from scratch. The experiment confirmed that the automatically generated model can also be efficiently used as a starting point for manual design of the target model, since it significantly shortens the time required for design. The calculated speed-up factors confirmed that the manual design, which takes the automatically generated model as a starting base, almost bisects the estimated efforts and actual time spent to obtain the target model in contrast to the manual design from scratch.

Some potential threats to validity of the experiment [19, 20] and derived conclusions are related to the source model quality. Someone may find that the used source BPM differs from the typical real BPMs, since it represents the result of a disciplined approach which forces modeling of resources. The approach is certainly dependent on the source

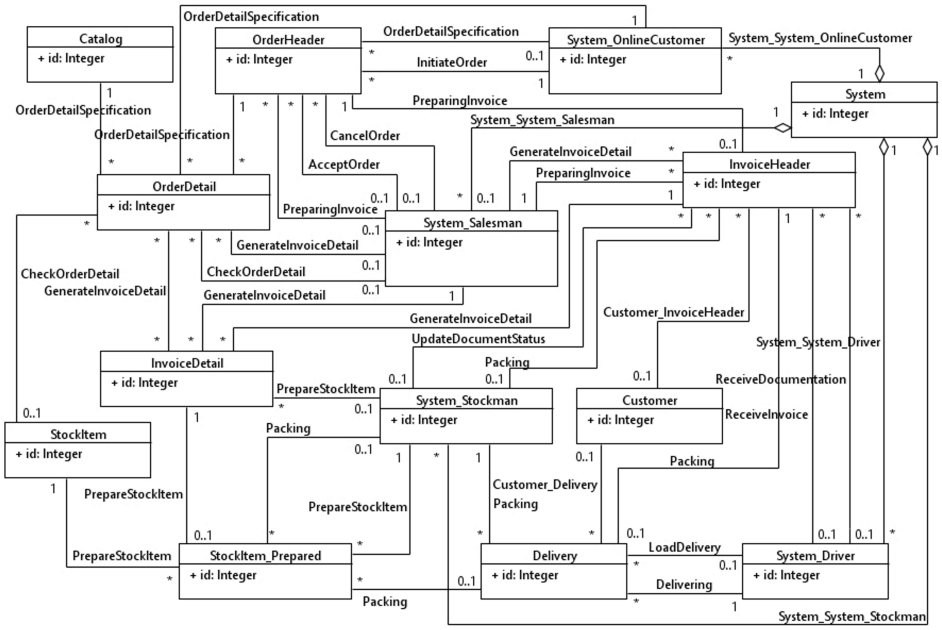


Fig. 15. UML class diagram representing the automatically generated CDM based on the BPMN model shown in Fig. 14

model quality, particularly on the representation of objects and object flows, since the completeness and correctness of the automatically generated CDM directly depend on the content of the source BPM. However, other approaches to automatic CDM synthesis are also dependent on the quality of the used source specification, regardless of whether it is textual or graphical. The experiment [19, 20] showed that a source BPM, as a result of the disciplined business modeling approach (respecting used objects and object flows), may constitute a reliable starting base for automatic CDM synthesis.

Apart from the assessment of the effectiveness and efficiency of the implemented tool, its usability could also be evaluated from the database designer's perspective. Currently we do not have any quantitative evaluation results, but we can compare the implemented system against the existing tools aimed to MDSDM. The implemented client application, which consumes the online CDM generator service, constitutes the first online publicly available tool for automatic BPM-driven CDM synthesis, which can be used for automatic synthesis of the initial CDM based on BPMs represented by two different concrete notations (BPMN and UML AD). After downloading the automatically generated CDM, a designer is able to use it in other modeling tools/platforms, without any installation and customization of additional tools and/or plugins in contrast to the existing approaches.

Validation from the developers' perspective is the process of checking whether the proposed approach and implemented online system (including all related services) satisfy the developers' needs.

In the context of the main goal of this research – development of an approach that enables implementation of an online service for automatic CDM synthesis based on BPMs represented by different concrete notations with the minimized dependency on the platform/vendor specificities, we can conclude that the implemented online service satisfies the developers' needs. Firstly, the proposed approach enables service-oriented architecture of the online system. Such a modular architecture enables separation of concerns and concurrent development of different services included in the orchestration, which further brings other related benefits. Secondly, the publicly available online CDM generator service enables other developers to simply consume it in their own applications and modeling platforms without any installation and customization of additional tools and/or plugins in contrast to the existing approaches. This could be very beneficial for researchers and other categories of developers.

10. Conclusions

In this article we presented an approach that enables automated CDM synthesis independently of different starting BPM notations. We identified BPM concepts having semantic potential for automated CDM synthesis, and we specified a simple DSL named BMRL for the representation of those characteristic concepts. With the introduction of DSL, the CDM synthesis is split into two phases. In the first phase, the specified concepts are extracted from the source BPM and represented by BMRL. In the second phase, the BMRL-based representation of the extracted BPM concepts is used for the automated generation of the target CDM. Each phase is based on a set of formal transformation rules enabling automatization of the whole process.

The proposed approach has several advantages over the existing approaches since it enables splitting of the CDM synthesis into two different phases. The first phase deals only with the extraction of the characteristic concepts from the source BPM independently of the target CDM synthesis, while the second phase only deals with the target CDM synthesis independently of the source BPM extraction. This approach reduces the CDM synthesis dependency on the source BPM notations that are caused by the metamodel changes and/or vendor specific implementations as well. If some source BPM notation is changed, then only the corresponding BPM extractor is to be changed. If some modifications of the generation rules are necessary, then only the CDM generator is to be modified, while the BPM extractors remain unchanged. Thus, the proposed approach facilitates the implementation of the required tools and simplifies the target CDM synthesis.

The proposed approach enables implementation of modular tools for BPM-driven CDM synthesis, which consist of loosely coupled components aimed at automatic extraction of specific concepts from the source models and automatic generation of the target model. Based on the proposed approach, we implemented the first online BPM-driven CDM generator as a web-based, platform- and source notation-independent tool. Currently, it enables automatic generation of the target CDM represented by the UML class diagram, based on BPMs represented by two concrete notations: BPMN and UML activity diagram. Its usage can be twofold. Firstly, database designers are able to use it through the implemented client application, which enables the source BPM upload and target CDM download – after downloading the automatically generated CDM, a

designer is able to use it in other modeling tools/platforms, without any installation and/or customization of additional tools/plugins in contrast to the existing approaches. Secondly, developers are able to consume the exposed web service from their own applications and modeling platforms, which could be very beneficial for researchers and other categories of developers.

Since the proposed approach to two-phase BPM-driven CDM synthesis is based on the identified semantic capacity of BPMs for the direct synthesis, which has previously been experimentally confirmed, the implemented online CDM generator is characterized by the same effectiveness and efficiency as the existing, experimentally evaluated, direct ATL-based generator. This means that the generator enables automatic generation of the target conceptual database model with very high completeness and precision: the average effectiveness was $\sim$80% for automatic generation of classes, and $\sim$85% for associations; the average recall of the generated model was above 80%; the average precision for automatically generated classes was above 75%, while the average precision for associations was about 90%. The experiments imply that the automatically generated model can also be efficiently used as a starting point for a manual design of the target model, since it significantly shortens the time required for design – the calculated speed-up factors confirm that the manual design, which takes the automatically generated model as a starting base, almost bisects the estimated efforts and actual time spent to obtain the target model in contrast to the manual design from scratch.

Our future work will focus on further identification of the semantic capacity of BPMs for automatic CDM synthesis, as well as improvements of the implemented tools, particularly BPM extractors in order to minimize vendor specificities. We also intend to provide visualization and editing functionalities of automatically generated models in the web browser.

References

1. Aguilar, J.A., Garrigós, I., Mazón, J.N., Trujillo, J.: An MDA approach for goal-oriented requirement analysis in web engineering. Journal of Universal Computer Science 16(17), 2475–2494 (2010)
2. Alencar, F., Marín, B., Giachetti, G., Pastor, O., Pimentel, J.H.: From i* requirements models to conceptual models of a model driven development process. In: Persson, A., Stirna, J. (eds.) POEM 2009, LNBIP, vol. 39, pp. 99–114. Springer, Berlin Heidelberg (2009)
3. Alencar, F., Pedroza, F., Castro, J., Amorim, R.: New mechanisms for the integration of organizational requirements and object oriented modeling. In: Proc. of WER 2003. pp. 109–123 (2003)
4. Alencar, F.M.R., Filho, G.A.C., Castro, J.F.: Support for structuring mechanism in the integration of organizational requirements and object oriented modeling. In: Proc. of WER 2002. pp. 147–161 (2002)
5. Alencar, F.M.R, Pedroza, F.P., Castro, J., Silva, C.T.L., Ramos, R.A.: XGOOD: A tool to automatize the mapping rules between i* framework and UML. In: Proc. of CIbSE 2006. pp. 125–138 (2006)
6. Ang, C.L., Khoo, L.P., Gay, R.K.L.: IDEF*: a comprehensive modelling methodology for the development of manufacturing enterprise systems. Int. Journal of Production Research 37(17), 3839–3858 (1999)
7. Banjac, D., Brdjanin, D., Banjac, G., Maric, S.: Evaluation of automatically generated conceptual database model based on collaborative business process model: Controlled experiment. In:

Stojanov, G., Kulakov, A. (eds.) ICT Innovations 2016, AISC, vol. 665, pp. 134–145. Springer (2016)
8. Becker, L.B., Pereira, C.E., Dias, O.P., Teixeira, I.M., Teixeira, J.P.: MOSYS: A methodology for automatic object identification from system specification. In: Proc. of ISORC 2000. pp. 198–201. IEEE Computer Society (2000)
9. Bloomfield, T.: MDA, meta-modelling and model transformation: Introducing new technology into the defence industry. In: Hartman, A., Kreische, D. (eds.) ECMDA-FA 2005, LNCS, vol. 3748, pp. 9–18. Springer, Berlin Heidelberg (2005)
10. Boccalatte, A., Giglio, D., Paolucci, M.: ISYDES: the project of a tool aimed at information system development. In: Proc. of AIWORC 2000. pp. 293–298. IEEE (2000)
11. Brambilla, M., Cabot, J., Comai, S.: Automatic generation of workflow-extended domain models. In: Engels, G., et al. (eds.) MoDELS 2007, LNCS, vol. 4735, pp. 375–389. Springer, Berlin Heidelberg (2007)
12. Brambilla, M., Cabot, J., Comai, S.: Extending conceptual schemas with business process information. Advances in Software Engineering, vol. 2010, Article ID 525121 (2010)
13. Brdjanin, D., Maric, S.: Towards the initial conceptual database model through the UML metamodel transformations. In: Proc. of Eurocon 2011. pp. 1–4. IEEE (2011)
14. Brdjanin, D., Maric, S.: An Approach to Automated Conceptual Database Design Based on the UML Activity Diagram. Computer Science and Information Systems 9(1), 249–283 (2012)
15. Brdjanin, D., Maric, S.: Model-driven Techniques for Data Model Synthesis. Electronics 17(2), 130–136 (2013)
16. Brdjanin, D., Maric, S., Gunjic, D.: ADBdesign: An approach to automated initial conceptual database design based on business activity diagrams. In: Catania, B., Ivanovic, M., Thalheim, B. (eds.) ADBIS 2010, LNCS, vol. 6295, pp. 117–131. Springer, Berlin Heidelberg (2010)
17. Brdjanin, D., Banjac, D., Banjac, G., Maric, S.: An approach to automated two-phase business model-driven synthesis of data models. In: Ouhammou, Y., et al. (eds.) Model and Data Engineering, LNCS, vol. 10563, pp. 57–70. Springer (2017)
18. Brdjanin, D., Banjac, D., Banjac, G., Maric, S.: An online business process model-driven generator of the conceptual database model. In: 8th International Conference on Web Intelligence, Mining and Semantics – WIMS'18. pp. 16:1–16:9. ACM (2018)
19. Brdjanin, D., Banjac, G., Banjac, D., Maric, S.: Controlled experiment in business model-driven conceptual database design. In: Reinhartz-Berger, I., et al. (eds.) Enterprise, Business-Process and Information Systems Modeling, LNBIP, vol. 287, pp. 289–304. Springer (2017)
20. Brdjanin, D., Banjac, G., Banjac, D., Maric, S.: An experiment in model-driven conceptual database design. Software & Systems Modeling pp. 1–25 (2018)
21. Brdjanin, D., Banjac, G., Maric, S.: Automated synthesis of initial conceptual database model based on collaborative business process model. In: Bogdanova, M.A., Gjorgjevikj, D. (eds.) ICT Innovations 2014: World of Data, AISC, vol. 311, pp. 145–156. Springer International Publishing, Cham (2015)
22. Brdjanin, D., Maric, S.: On automated generation of associations in conceptual database model. In: De Troyer, O., et al. (eds.) ER Workshops 2011, LNCS, vol. 6999, pp. 292–301. Springer-Verlag, Berlin Heidelberg (2011)
23. Brdjanin, D., Maric, S.: Towards the automated business model-driven conceptual database design. In: Morzy, T., Harder, T., Wrembel, R. (eds.) Advances in Databases and Information Systems, AISC, vol. 186, pp. 31–43. Springer-Verlag, Berlin Heidelberg (2012)
24. Budinsky, F., Steinberg, D., Merks, E., Ellersick, R., Grose, T.: Eclipse Modeling Framework. Pearson Education, Boston, USA (2003)
25. Castro, J.F., Alencar, F.M.R., Filho, G.A.C., Mylopoulos, J.: Integrating organizational requirements and object oriented modeling. In: Proc. of ISRE 2001. pp. 146–153. IEEE (2001)
26. Cruz, E.F., Machado, R.J., Santos, M.Y.: From business process modeling to data model: A systematic approach. In: Proc. of QUATIC 2012. pp. 205–210. IEEE (2012)

27. Cruz, E.F., Machado, R.J., Santos, M.Y.: Deriving a data model from a set of interrelated business process models. In: Proc. of ICEIS 2015. pp. 49–59 (2015)
28. de la Vara, J.L.: Business process-based requirements specification and object-oriented conceptual modelling of information systems. PhD Thesis, Valencia Polytechnic Uni. (2011)
29. Drozdova, M., Kardos, M., Kurillova, Z., Bucko, B.: Transformation in model driven architecture. In: Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I. pp. 193–203. Springer, Cham (2016)
30. Drozdová, M., Mokryš, M., Kardoš, M., Kurillová, Z., Papán, J.: Change of paradigm for development of software support for elearning. In: Proc. of ICETA 2012. pp. 81–84. IEEE (2012)
31. España, S.: Methodological integration of communication analysis into a model-driven software development framework. PhD Thesis, Valencia Polytechnic Uni. (2011)
32. Essebaa, I., Chantit, S.: Toward an automatic approach to get pim level from cim level using qvt rules. In: 2016 11th International Conference on Intelligent Systems: Theories and Applications (SITA). pp. 1–6. Mohammedia (2016)
33. Fernandes, J.M., Lilius, J., Truscan, D.: Integration of DFDs into a UML-based model-driven engineering approach. Software and Systems Modeling 5(4), 403–428 (2006)
34. Fouad, A.: Embedding requirements within the model driven architecture. PhD Thesis, Bournemouth Uni. (2011)
35. Insfran, E., Pastor, O., Wieringa, R.: Requirements Engineering-Based Conceptual Modelling. Requirements Engineering 7(2), 61–72 (2002)
36. Insfran, E.: Requirements engineering approach for object-oriented conceptual modeling. PhD Thesis, Valencia Polytechnic Uni. (2003)
37. Jiang, L., Topaloglou, T., Borgida, A., Mylopoulos, J.: Goal-oriented conceptual database design. In: Proc. of RE '07. pp. 195–204. IEEE, Los Alamitos, USA (2007)
38. Jouault, F., Allilaire, F., Bezivin, J., Kurtev, I.: ATL: A model transformation tool. Science of Computer Programming 72(1-2), 31–39 (2008)
39. Koch, N.: Transformation Techniques in the Model-Driven Development Process of UWE. In: Proc. of the Workshops at ICWE'06, Art. No. 3. ACM (2006)
40. Koch, N., Zhang, G., Escalona, M.J.: Model Transformations from Requirements to Web System Design. In: Proc. of ICWE'06. pp. 281–288. ACM (2006)
41. Koskinen, J., Peltonen, J., Selonen, P., Systa, T., Koskimies, K.: Model processing tools in UML. In: Proc. of ICSE 2001. pp. 819–820. IEEE Computer Society (2001)
42. Kriouile, A., Addamssiri, N., Gadi, T.: An MDA Method for Automatic Transformation of Models from CIM to PIM. American Journal of Software Engineering and Applications 4(1), 1–14 (2015)
43. Lingzhi, L., Ang, C.L., Gay, R.K.L.: Integration of Information Model (IDEF1) with Function Model (IDEF0) for CIM Information System Design. Expert Systems with Applications 10(3/4), 373–380 (1996)
44. Liu, D., Subramaniam, K., Far, B., Eberlein, A.: Automating Transition from Use-cases to Class Model. In: Proc. of CCECE 2003. pp. 831–834. IEEE (2003)
45. Martinez Rebollar, A.: Conceptual schemas generation from organizational models in an automatic software production process. PhD Thesis, Valencia Polytechnic Uni. (2008)
46. Nikiforova, O., Gusarovs, K., Gorbiks, O., Pavlova, N.: BrainTool: A tool for generation of the UML class diagrams. In: Proc. of ICSEA 2012. pp. 60–69. IARIA (2012)
47. Nikiforova, O., Gusarovs, K., Gorbiks, O., Pavlova, N.: Improvement of the two-hemisphere model-driven approach for generation of the uml class diagram. Applied Computer Systems 14(1), 19–30 (2013)
48. Nikiforova, O., Pavlova, N.: Application of BPMN instead of GRAPES for two-hemisphere model driven approach. In: Grundspenkis, J., et al. (eds.) ADBIS 2009 Workshops, LNCS, vol. 5968, pp. 185–192. Springer, Berlin Heidelberg (2010)

49. OMG: MOF 2.0 Query/View/Transformation Specification, v1.0. OMG (2008)
50. OMG: Business Process Model and Notation (BPMN), v2.0. OMG (2011)
51. OMG: Unified Modeling Language (OMG UML), v2.5. OMG (2015)
52. Rodriguez, A., Fernandez-Medina, E., Piattini, M.: Analysis-level classes from secure business processes through model transformations. In: Lambrinoudakis, C., Pernul, G., Tjoa, A.M. (eds.) TrustBus 2007, LNCS, vol. 4657, pp. 104–114. Springer, Berlin Heidelberg (2007)
53. Rodriguez, A., Fernandez-Medina, E., Piattini, M.: Towards obtaining analysis-level class and use case diagrams from business process models. In: Song, I.Y., et al. (eds.) ER Workshops 2008, LNCS, vol. 5232, pp. 103–112. Springer, Berlin Heidelberg (2008)
54. Rodriguez, A., Garcia-Rodriguez de Guzman, I., Fernandez-Medina, E., Piattini, M.: Semi-formal transformation of secure business processes into analysis class and use case models: An MDA approach. Information and Software Technology 52(9), 945–971 (2010)
55. Rungworawut, W., Senivongse, T.: Using ontology search in the design of class diagram from business process model. PWASET 12, 165–170 (2006)
56. Santos, M.Y., Oliveira e Sá, J.: A Data Warehouse Model for Business Processes Data Analytics, pp. 241–256. Springer International Publishing, Cham (2016)
57. Santos, M.Y., Machado, R.J.: On the derivation of class diagrams from use cases and logical software architectures. In: Proc. of ICSEA '10. pp. 107–113. IEEE (2010)
58. Selonen, P., Koskimies, K., Sakkinen, M.: Transformations Between UML Diagrams. Journal of Database Management 14(3), 37–55 (2003)
59. Silva, L.F., Leite, J.C.S.P.: Generating requirements views: A transformation-driven approach. Electronic Communications of the EASST 3, 1–14 (2006)
60. Srivastava, S.: Model transformation approach for a goal oriented requirements engineering based webgrl to design models. International Journal of Soft Computing and Engineering (IJSCE) 3(6), 66–75 (2014)
61. Tan, H.B.K., Yang, Y., Blan, L.: Systematic transformation of functional analysis model in object oriented design and implementation. IEEE Transaction on Software Engineering 32(2), 111–135 (2006)
62. Truscan, D., Fernandes, J.M., Lilius, J.: Tool support for DFD-UML based transformation. In: Proc. of ECBS '04. pp. 378–387. IEEE (2004)
63. Voelter, M., Benz, S., Dietrich, C., Engelmann, B., Helander, M., Kats, L., Visser, E., Wachsmuth, G.: DSL Engineering – Designing, Implementing and Using Domain-Specific Languages (2013)
64. Wrycza, S.: The ISAC-driven transition between requirements analysis and ER conceptual modelling. Information Systems 15(6), 603–614 (1990)
65. Zhang, J., Feng, P., Wu, Z., Yu, D., Chen, K.: Activity based CIM modeling and transformation for business process systems. International Journal of Software Engineering and Knowledge Engineering 20(3), 289–309 (2010)

Drazen Brdjanin is an Associate Professor at the Faculty of Electrical Engineering, University of Banja Luka (Bosnia and Herzegovina), where he heads the M-lab Research Group. His research interests are mainly related to databases and model-driven software development. He was participating in several R&D projects at national and international level, and authoring a number of research papers and articles in the field of model-driven development and database design.

Danijela Banjac is a Senior Teaching Assistant and PhD student at the Faculty of Electrical Engineering, University of Banja Luka (Bosnia and Herzegovina). She is a member

of M-lab Research Group. Her research interests include model-driven software development, business process modelling, object-oriented information systems, and UML. She has published several research papers and articles.

Goran Banjac is a Senior Teaching Assistant and PhD student at the Faculty of Electrical Engineering, University of Banja Luka (Bosnia and Herzegovina). He is a member of M-lab Research Group. His research interests include model-driven software development, business process modelling, databases, and UML. He has published several research papers and articles.

Slavko Maric is a Full Professor at the Faculty of Electrical Engineering, University of Banja Luka (Bosnia and Herzegovina), where he heads the Computer Science Department. His current research interests include: information systems modeling, design and development, databases, eGovernment systems, service oriented architecture and parallel processing. He has published over 50 research papers and articles, and participated in a number of research and development projects.

Received: October 10, 2018; Accepted: June 2, 2019.

CIP – Каталогизација у публикацији
Народна библиотека Србије, Београд

004

COMPUTER Science and Information Systems : the International journal /
Editor-in-Chief Mirjana Ivanović. – Vol. 16, No 2 (2019) - . – Novi Sad (Trg D. Obradovića 3): ComSIS Consortium, 2018 - (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