JYVÄSKYLÄ STUDIES IN COMPUTING
189

Mikhail Zolotukhin

On Data Mining Applications in Mobile Networking and Network Security

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 189

Mikhail Zolotukhin

On Data Mining Applications in Mobile Networking and Network Security

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston vanhassa juhlasalissa S212
toukokuun 22. päivänä 2014 kello 12.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in building Seminarium, auditorium S212 on May 22, 2014 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2014

On Data Mining Applications in Mobile Networking and Network Security

JYVÄSKYLÄ STUDIES IN COMPUTING 189

Mikhail Zolotukhin

On Data Mining Applications in Mobile Networking and Network Security

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2014

Editors
Timo Männikkö
Department of Mathematical Information Technology, University of Jyväskylä
Pekka Olsbo, Ville Korkiakangas
Publishing Unit, University Library of Jyväskylä

URN:ISBN:978-951-39-5676-9
ISBN 978-951-39-5676-9 (PDF)

ISBN 978-951-39-5675-2 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2014

ABSTRACT

Zolotukhin, Mikhail
On data mining applications in mobile networking and network security
Jyväskylä: University of Jyväskylä, 2014, 72 p.(+included articles)
(Jyväskylä Studies in Computing
ISSN 1456-5390; 189)
ISBN 978-951-39-5675-2 (nid.)
ISBN 978-951-39-5676-9 (PDF)
Finnish summary
Diss.

This work focuses on the application of different methods and algorithms of data mining to various problems encountered in mobile networks and computer systems. Data mining is the process of analysis of a dataset in order to extract knowledge patterns and construct a model for further use based on these patterns. This process involves three main phases: data preprocessing, data analysis and validation of the obtained model. All these phases are discussed in this study. The most important steps of each phase are presented and several methods of their implementation are described. In addition, several case studies devoted to different problems in the field of computer science are presented in the dissertation. Each of these studies employs one or more data mining techniques to solve a posed problem. Firstly, optimal positions of relay stations in WiMAX multihop networks are calculated with the help of genetic algorithm. Next, the prediction of the next mobile user location is carried out based on the analysis of spatial-temporal trajectories and application of several classifying methods. After that, the use of clustering and anomaly detection techniques for the detection of anomalous HTTP requests is presented. Finally, the data mining approach is applied for the detection and classification of malicious software. These case studies show that data mining methods can help to solve many different problems related to mobile networking and network security.

Keywords: Data mining, machine learning, classification, clustering, anomaly detection, relay station, mobile data, network security

Author

Mikhail Zolotukhin
Department of Mathematical Information Technology
University of Jyväskylä
Finland

E-mail: `mikhail.zolotukhin@jyu.fi`

Supervisor

Professor Timo Hämäläinen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers

Professor János Sztrik
Department of Informatics Systems and Networks
University of Debrecen
Hungary

Associate Professor Dmytro Andrushko
Telecommunicational Systems Department
Kharkov National University of Radioelectronics
Ukraine

Opponent

Research Professor Ilkka Norros
VTT Technical Research Centre of Finland
Finland

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Prof. Timo Hämäläinen for his guidance and support. His help and encouragement have been invaluable.

I am indebted to my collaborators, Prof. Andrey Garnaev, Mr. Vesa Hytönen, Dr. Oleksandr Puchko, Mr. Antti Juvonen, Mr. Thomas Höhne, Mr. Tom Chapman, Ms. Elena Ivannikova, and Prof. Victor Zakharov. Also I would like to thank Dr. Oleksandr Sayenko for providing interesting tasks and ideas and Dr. Steve Legrand for the help in organizing all of the articles included into this dissertation.

I am grateful to the Faculty of Information Technology of the University of Jyväskylä for giving me the opportunity to work on my dissertation. During my studies, I participated in and was funded by the programme offered by the Graduate School in Electronics, Telecommunications and Automation (GETA), and I would like to thank all the people related to this programme. In addition, I am thankful to the Jyväskylä Doctoral Program in Computing and Mathematical Sciences (COMAS) and Sami Kollanus for supporting me in the early stages of my research.

Finally, I would like to thank my family and friends who have given me a perspective to life beyond the academic world.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

LIST OF INCLUDED ARTICLES

PI Mikhail Zolotukhin, Vesa Hytönen, Timo Hämäläinen and Andrey Garnaev. Optimal Relays Deployment for 802.16j Networks. *Springer. Mobile Networks and Management. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Vol. 97, pp. 31–45, 2012.

PII Mikhail Zolotukhin, Timo Hämäläinen and Andrey Garnaev. A Relay Deployment Mechanism for One Scenario of Cost-Effective Coverage Extension in IEEE 802.16j Networks. *Proceedings of the 5th International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–6, 2012.

PIII Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen. Online anomaly detection by using N-gram model and growing hierarchical self-organizing maps. *Proceedings of the 8th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 47–52, 2012.

PIV Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen. Growing Hierarchical Self-organizing Maps and Statistical Distribution Models for Online Detection of Web Attacks. *Springer. Web Information Systems and Technologies. Lecture Notes in Business Information Processing*, Vol. 140, pp. 281–295, 2013.

PV Mikhail Zolotukhin, Elena Ivannikova and Timo Hämäläinen. Novel Method for the Prediction of Mobile Location Based on Temporal-Spatial Behavioral Patterns. *Proceedings of the 3rd IEEE International Conference on Information Science and Technology (ICIST)*, pp. 761–766, 2013.

PVI Mikhail Zolotukhin and Timo Hämäläinen. Detection of Anomalous HTTP Requests Based on Advanced N-gram Model and Clustering Techniques. *Springer. Internet of Things, Smart Spaces, and Next Generation Networking. Lecture Notes in Computer Science*, Vol. 8121, pp. 371–382, 2013.

PVII Mikhail Zolotukhin and Timo Hämäläinen. Support Vector Machine Integrated with Game-Theoretic Approach and Genetic Algorithm for the Detection and Classification of Malware. *Proceedings of Globecom 2013 Workshop - the 1st International Workshop on Security and Privacy in Big Data*, pp. 211–216, 2013.

PVIII Mikhail Zolotukhin and Timo Hämäläinen. Detection of Zero-day Malware Based on the Analysis of Opcode Sequences. *Proceedings of the 11th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, pp. 677–682, 2014.

1 INTRODUCTION

This chapter presents the motivation behind the research devoted to the application of different data mining algorithms, which are used for solving several problems in mobile networks and network security. Next, the research topics are outlined and research approach is briefly described. Following the presentation of the structure of the dissertation, the articles included in this dissertation and the author's contribution to them are individually introduced.

1.1 Research motivation

Recent growth in the use of computer technologies, both for work and personal use, has led to the development of new means to automatically gather huge volumes of diverse data. The analysis of the collected data is supposed to help in solving problems related to behavior and interactions of humans and machines. An example of such data could be users locations that a mobile service provider is able to collect with the help of their mobile communication devices. Afterwards, the provider can use these data to optimize the allocation of the network's limited resources and ensure that the network service is available anywhere and anytime [AGA10]. Another example is statistics of network traffic sent to and received by a web server and gathered by a special computer software installed on this server. The analysis of the gathered data can help the system administrator to prevent users' activity that can corrupt the server or be used to collect confidential information [PJ13]. System call sequences issued by the executed software, their parameters and their return values can be collected by a special computer program. After analyzing the collected statistics, the program can potentially detect computer viruses and malicious software by their suspicious behavior [FSHS10].

These problems are connected with each other by the fact that all of them rely on the analysis and the discovery of patterns from large amounts of data of different types: text, numeric and boolean. In computer science, this analysis process is known as data mining, and it involves database and data management

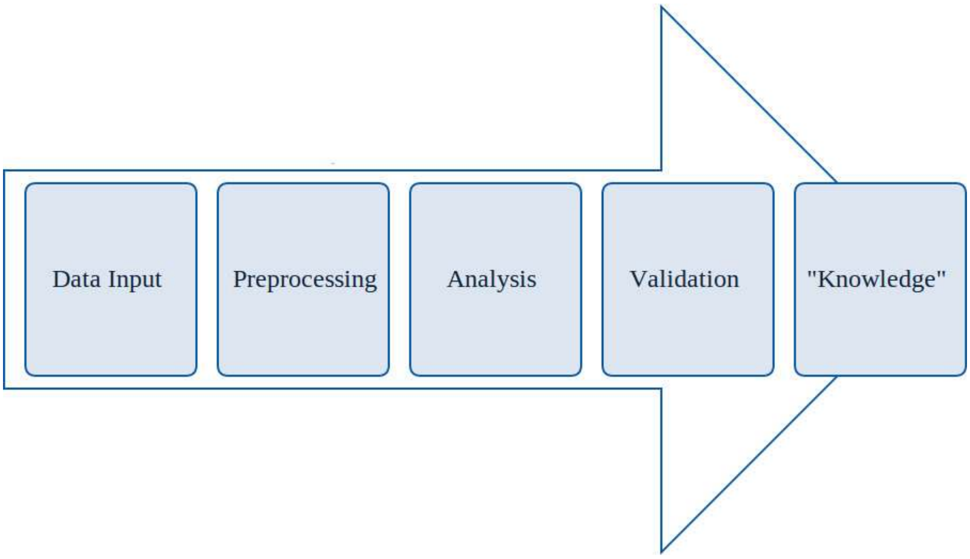


FIGURE 1 The scheme of the data mining process

aspects, data preprocessing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating [CEF$^+$06]. The main aim of the data mining process is to extract information from a dataset and transform it into an understandable structure for further use [PSF91, FPSS96, Loh12]. Nowadays, different data mining methods and algorithms are employed, in a varying degree, in business [PTC13] and decision making [BP10], medicine [XKB$^+$05] and biology [CHS$^+$01], software engineering [TTLC09] and network security [LDb10].

As shown in Figure 1, the data mining process for knowledge extraction from input data usually includes three main steps: data preprocessing, data analysis and result validation [DYY13]. Preprocessing is an essential step in analyzing the multivariate datasets before data mining algorithms can be used. This step can include the extraction of necessary statistics from the raw data, the selection of the most essential features and dimensionality reduction, and the standardization and normalization. During this step, the most effective data transformations are supposed to be carried out so that a data analysis algorithm can work efficiently. In addition, these transformations must properly encode a priori knowledge of the problem.

During the data analysis step, an appropriate data mining algorithm should be chosen to build a model which accurately describes the structure of the data and allows one to extract required information from the dataset. Data mining involves six common classes of tasks [FPSS96]: anomaly detection [CBK09], association rule learning [HGN00], clustering [Jai10], classification [SSB11], regression [KP09], and summarization [RS02]. On this stage, an algorithm is supposed to be selected carefully according to the posed problem, and its implementation must take into account the specifics of the dataset.

The final step of the data mining process is to verify that the patterns produced on the data analysis stage occur in the wider dataset. For this purpose, the result validation process uses a test set of data on which the data mining algorithm was not trained. The learned patterns are applied to this test set, and the result is compared to the desired output.

The resulting "knowledge" extracted during the data mining process includes previously unknown patterns such as clusters of data samples, anomalous data structures, or dependencies between data features. These patterns form a model that describes the structure of the analyzed data. Furthermore, this model can be applied to analyze sets that contain data of similar kind.

In mobile networks, data mining schemes can be used to evaluate a network performance, find unexpected delays and system faults [LR05], predict bandwidth capacity requirements, allocate network's limited resources based on the analysis of time-series extracted from historical traffic data [GX09], and provide location-based recommendations and advertisements by exploring the mobile user's previous moves and current context [GTL12]. Data mining algorithms have also many applications in computer network security. They are applied to classify web-based attacks by analyzing logs of web servers [JS12], cluster network traffic based on packet statistics [LK12], detect computer viruses and trojans by exploring binary and opcode sequences extracted from files [SBUPB13] or by evaluating a software system calls [FSHS10], and find and block spam emails and messages with the help of various textual mining techniques [AAN11].

1.2 Research topics

The objective of the research is to apply different methods and schemes of data mining to various problems encountered in mobile networking and network security. To achieve this, several case studies are presented in this dissertation. In each of these studies, one or more data mining techniques are employed in order to solve a posed problem. All issues considered in the studies can be divided into the following groups:

1. Search for optimal positions of relay stations in WiMAX multi-hop networks with the help of a genetic algorithm.
2. Prediction of the next user location based on the analysis of spatial-temporal trajectories.
3. Application of clustering and anomaly detection techniques to identify intrusive HTTP requests.
4. Data mining approach for the detection of known and zero-day malware.

1.3 Research approach

The techniques and methods used in the research described in this dissertation can be referred to as the Constructive Research Approach (CRA). This approach aims to create an innovative artifact in order to solve a real-world problem and therefore to contribute to the field of study where it has been applied [PG13]. Here artifact may denote a model, chart, plan or strategy, organizational structure, commercial product, or information system.

The CRA process starts by finding or choosing a scientifically relevant real-world problem. The second step is the analysis of this problem and review of relevant literature to gain a holistic and thorough understanding of the problem and find existing methods for its solution. The third phase is the construction of an innovative artifact which is intended to solve the problem and supposed to be strongly connected to existing theoretical knowledge or practical experience. On the next step, CRA implies implementation of the artifact in order to test its functionality and applicability. The last phase of the process consists of observation of the outcome of the artifact installation and comparison of this outcome with the results observed before the artifact was introduced. Positive results can lead to the development of a new theory or the refinement of an existing one.

This approach is widely applied to data mining tasks. In such tasks, a problem refers to the analysis of data obtained from a real-world system and aims to extract knowledge patterns from these data. In this case, an artifact is a novel data mining algorithm or a modification of an existing one that takes into account the specifics of the problem. The algorithm is implemented as a computer program and applied to a test dataset to estimate the algorithm performance and to compare it with existing analogues. This comparison forms part of the theoretical contribution.

1.4 Structure of the work

The rest of this thesis is organized as follows. First, the theoretical background of several approaches and algorithms of data mining is discussed. Second, the contribution of research articles and case studies is presented. The last part draws the conclusions and outlines future work.

In Chapter 2, the process of data mining is introduced. The main steps of data preprocessing are presented. These include feature extraction and selection, standardization, normalization and dimensionality reduction. Next, such data mining tasks as classification, clustering and anomaly detection are discussed. In addition, several schemes and methods for data preprocessing and analysis are described in more detail. Finally, techniques for algorithm validation and performance evaluation are shown.

Chapter 3 outlines the research contribution of the dissertation. It focuses on the use of different data mining algorithms to solve several problems encountered in the field of mobile networks and computer network security. First, each of these problems is briefly introduced. Next, the approach for the problem solution is described. Finally, the results of numerical simulations are shown.

Chapter 4 concludes the dissertation and briefly outlines the author's future steps directed to solve not only issues presented in Chapter 3 but also other problems found in computer networks and systems, for which the application of data mining methods can be useful.

1.5 Author's contribution to the included articles

The author's contribution to the included articles is in the development of the entire framework required to find a solution of each problem posed. Such framework involves the problem formulation, the development of a method for its solution, the evaluation of the proposed mechanism performance, and its comparison with existing analogues. Articles [PI, PII] present the problem of optimal relay station deployment for the IEEE 802.16j networks. In article [PV], next user location is predicted based on the analysis of spatial-temporal trajectories. In [PIII, PIV, PVI], the problem of online detection of anomalous HTTP requests is considered. Finally, in order to solve the problem of malware detection and classification, articles [PVII, PVIII] apply an approach based on data mining.

In [PI], the problem of calculation of optimal positions of relay stations in a IEEE 802.16j network is formulated. A region which already contains several base stations is considered. The aim is to deploy relay stations in a way that allows maximization of the signal-to-noise ratio for every user in the cell, taking into account the interference increase caused by the deployment of relays. The author formulates this problem as a nonlinear, non-convex and non-separable integer programming problem, solves it by using a genetic algorithm and carries out numerical calculations and network performance simulations. The proposed algorithm also allows us to find the optimal number of relay stations which should be placed in the region to maximize the available bandwidth.

The deployment of relay stations is one of the most promising technologies, not only to improve the signal-to-noise ratio for users in poor radio conditions but also to extend the service area. Article [PII] considers an optimization problem of cost-effective relay station deployment for the IEEE 802.16j networks in a case where the coverage is extended on a new residential area. The author calculates the provider's profit as a function of the network coverage area and tries to maximize it by finding an optimal number of relay stations and their optimal positions with the help of a genetic algorithm. The model also takes into account the costs caused by the deployment of relay stations. Finally, the author presents a numerical example of the proposed relay deployment mechanism, and the results show that the provider can drastically increase its own profit by using this method.

Article [PV] presents the problem of a mobile user's next location prediction based on periodic patterns learned from the previous user's moves and current context. The main aim is to employ a reliable user-specific approach which allows one to analyze spatial-temporal trajectories of a user and determine the next user location depending on his/her current location and time. The author reduces this problem to a classification problem with several specifics and solves it with the help of several classifying algorithms. The proposed scheme for the solution is tested using real data collected from the mobile phones of several non-related users over periods of time varying from a few weeks to two years. The simulation results show that the next user location can be predicted with an accuracy rate that is higher than of the existing analogues.

In Article [PIII], online detection of intrusive HTTP requests is described. For this purpose, an approach based on anomaly detection is employed. Such approach learns the features of event patterns corresponding to normal behavior and classifies patterns that deviate from the established norms as intrusions, therefore being able to detect zero-day attacks. The author's contribution involves the transformation of HTTP requests to numeric vectors by applying an n-gram model and the application of adaptive growing hierarchical self-organizing maps to find anomalies among these vectors. The technique proposed is self-adaptive and allows the detection of online HTTP attacks in the case of continuously updated web applications. Finally, the method is tested using logs which include normal and intrusive requests. As a result, almost all attacks recorded on these logs are detected while the number of false alarms remains very low.

Article [PIV] extends the results obtained in [PIII]. As previously, in this study the approach based on anomaly detection is used to find anomalous HTTP requests. In addition to the detection of intrusive HTTP queries by growing hierarchical self-organizing maps, the author employs a statistical distribution model to find code injections in HTTP headers. This model relies on the analysis of the lengths of header values and relative frequencies of non-alphanumeric symbols. The proposed approach allows one to detect various kinds of attacks based on HTTP response splitting or malicious redirecting in the case of continuous updated web-applications. To test the algorithm, the author uses logs and included normal and intrusive requests acquired from a large real-life web service. Simulation results show that almost all attacks from these logs are detected.

In article [PVI], the author continues to apply the anomaly detection based approach for the detection of intrusive HTTP requests. The study focuses on an algorithm processing HTTP queries, which allows the analysis of all HTTP request messages at once and does not separate them by resource. The author proposes a technique that takes into account combinations of non-alphanumeric symbols and frequencies of appearance of n-grams in each HTTP query separately and in the whole server log. After this preprocessing has been carried out, standard clustering techniques are employed to find anomalous HTTP requests. The algorithm proposed can be used to quickly build the model of normal behavior, even if the set of HTTP requests free of attacks cannot be extracted.

In [PVII], an approach based data mining, for both the detection of malware and its classification, is employed. The proposed approach relies on supervised machine-learning. The author works with executable files presented in the form of byte and opcode sequences and employs n-gram models to extract features from these sequences. A genetic algorithm is used to select the most essential features and a classification model is then built with the help of support vector machines. The author considers the problem of the combined classifiers as a decision-making task and applies game theory methods to detect executable files infected with a malware. Numerical examples show that the algorithm produces good results in terms of malware detection and classification accuracy.

The problem of malware detection is also presented in [PVIII]. As previously, the author contributes to the analysis of operation code sequences extracted from executable files, to the application of n-gram models to discover essential features from these sequences and to the construction of a benign software model to detect malicious executables within new files. This model is based on the iterative usage of support vector machines and support vector data descriptions. The scheme proposed allows one to detect malware unseen previously, and the simulation results show that the method results in a higher accuracy rate than that of the existing analogues.

1.6 Other published articles

In addition to the included articles, the author has also participated in the research of advanced receivers for high-speed single frequency network (HS-SFN) in High-Speed Downlink Packet Access (HS-DPA) and spectrum bandwidth management between spectrum holder and primary and secondary users. The results have been published in the following articles:

- O. Puchko, M. Zolotukhin, V. Hytonen, T. Hohne and T. Chapman. Enhanced LMMSE equalizer for high-speed single frequency network in HS-DPA. In Proc. of the Communication Technologies Workshop (Swe-CTW), pp. 92–97, 2011.
- O. Puchko, M. Zolotukhin, T. Hohne, T. Chapman and V. Hytonen. Phase Adjustment in HS-SFN for HSDPA. In Proc. of the 5th International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5, 2012.
- M. Zolotukhin, A. Garnaev and T. Hämäläinen. A Stackelberg tariff game between a provider, a primary and a secondary user. In Proc. of the 5th International Conference on Game Theory and Management (GTM), pp. 344–358, 2011.

From a mathematical point of view, the studies described in these papers are reduced to different optimization problems. Optimization is also a key point of many data mining algorithms. Thus, mathematical techniques used to find a solution for the problems presented in these studies might be similar to the methods applied during the data mining process.

2 THEORETICAL FOUNDATION

In this chapter, the data mining process is discussed in more details. The chapter mostly focuses on methods and algorithms employed in the articles included in this dissertation. First, the preprocessing step is outlined and its most important parts such as feature extraction, normalization and selection are described. Next, such data mining tasks as classification, clustering and anomaly detection are presented. Finally, metrics for a result evaluation are introduced and validation techniques are described.

2.1 Data preprocessing

As a rule, data preprocessing is an initial step in the data mining process. Due to the fact that data gathering methods are often loosely controlled, the resulting data can contain out-of-range values, impossible data combinations and missing values, among other things. The main aim of the preprocessing is to remove irrelevant and redundant information present in the data. The deletion of noisy and unreliable data allows the extraction of knowledge from the data more accurately and reduces processing time during later phases of the data mining process. Data preprocessing can include feature extraction and selection, standardization and normalization, and transformation and dimensionality reduction. The resulting product of data preprocessing is a training set from which knowledge can be discovered [KKP06].

2.1.1 Feature extraction

After data has been gathered, it is supposed to be transformed into a reduced representation set of features. During the feature extraction process, the relevant information from the input data is extracted. It allows us to perform the desired task by using the reduced representation instead of the full-size raw data.

Features extracted from a dataset can be of different types. As a rule, all feature types can be divided into two groups: discrete and continuous. Discrete features include nominal, which have finite or countably infinite sets of values and provide only information to distinguish one object from another, and ordinal, which provide enough information to order objects. Continuous features can be separated to two types: interval and ratio. For interval features, only the differences between values are meaningful, while, for ratio features, both differences and ratios are meaningful.

The method applied to extract essential features from a raw dataset strongly depends on the given task. For example, in order to analyze network traffic and detect anomalous connections from each network flow, IP address and port of the source and the destination host as well as number of transferred bytes can be extracted [NLLS11]. To solve a face recognition problem, the features extracted from a face image can involve coordinates of eyes, nasal wings and corners of the mouth [RHB11]. States and transitions of a deterministic finite automaton which goes through tokens generated by an incoming HTTP request and adds a new transition and a new state when encountering a previously unseen token can be employed to detect anomalous HTTP requests [LDb10].

2.1.1.1 N-gram

One method which is widely used in the articles included in this thesis is based on the application of n-gram models. Such models are widely used in statistical natural language processing [Sue79] and speech recognition [HPK09]. An n-gram is a sub-sequence of n overlapping items (characters, letters, words) from a given sequence. For example, the 2-gram word model applied to the string "this is a feature" gives the following result: (“this“, ”is”), (“is”, “a”), (“a”, “feature”). In language modeling, to simplify the problem of learning the language model from data, it is assumed that each word depends only on the last $n - 1$ words and, therefore, an n-gram model can be applied as an approximation of the true underlying language.

Various techniques based on the application of n-gram models are often used to extract features from textual and numeric data. In study [LWSH05], an n-gram model is applied to analyze binary content of files and to categorize their type. An n-gram distribution is computed by sliding a fixed-size window through the set of data and counting the number of occurrences of each gram. In order to detect anomalous programs, study [HBN11] uses n-gram modeling of short sequences of system calls along with the occurrence frequency in the training traces. In [JS12], each HTTP request is first transformed to several sequences of n characters with the help of an n-gram model, and then these sequences are used to construct a vector, which expresses the frequency of every n-character in the analyzed request.

2.1.2 Standardization and normalization

Normalization is one of the first steps supposed to be applied to the continuous features extracted from a dataset. This step is very important when dealing with parameters of different units and scales. For example, if a data mining technique uses the Euclidean distance, all features should have the same scale for a fair comparison between them.

There are various methods available to perform data normalization. For example, min-max normalization performs a linear alteration on the original data so that the values are normalized within the given range. To map a value x_i of an attribute $x = (x_1, x_2, \dots, x_n)$ from range $[x_{min}, x_{max}]$ to a new range $[\bar{x}_{min}, \bar{x}_{max}]$, the computation is carried out as follows

$$\bar{x}_i = \bar{x}_{min} + \frac{x_i - x_{min}}{x_{max} - x_{min}}(\bar{x}_{max} - \bar{x}_{min}), \tag{1}$$

where $\bar{x}_i$ is the new value in the required range [SM09]. The benefit of min-max normalization is that all the values are mapped into a certain range. On the other hand, if a dataset contains outliers, min-max normalization will scale the normal data to a very small interval, which can cause difficulties during the analysis stage.

Another well-known normalization technique is referred to as z-score normalization or standardization. This approach normalizes a feature x based on the mean μ and standard deviation σ of its values:

$$\bar{x}_i = \frac{x_i - \mu}{\sigma}, \tag{2}$$

where $\bar{x}_i$ is the new value of x_i [SM09]. The absolute value of $\bar{x}_i$ represents the distance between the raw value x_i and the feature's mean in units of the standard deviation. Z-score normalization allows us to compare disparate measures. However, it assumes that the data have been generated with the Gaussian law. If it is not the case in reality, the representation of the data can be spoiled after the application of z-score standardization.

2.1.3 Feature selection

The number of extracted features can be huge, which may result in the use of large amounts of computing and memory resources when applying a data mining algorithm. Moreover, some of those features cannot provide any useful information in any context. Feature selection is the process of selecting a subset of the most relevant features for use in model construction. Feature selection allows us to improve model interpretability, reduce training times and enhance generalization by reducing overfitting. An optimal feature subset can be generated with the help of an algorithm of global search for the most relevant features. Another option is to sequentially add and remove items from a subset of features until the most optimal subset is found. The optimality criterion might be based on correlation, mutual information, error probability, or inter-class distance [MBN02].

2.1.3.1 Relief

RELIEF is a feature selection algorithm widely used in classification. The algorithm requires that data contained in the training dataset has been categorized [KR92]. It is noise-tolerant and robust to feature interactions and can be applied for discrete or continuous data.

The key idea of the method is to estimate features according to how well their values distinguish among instances that are near each other. For this purpose, for each feature i a weight w_i equal to zero is assigned. Then, for a feature vector $x = \{x_1, x_2, \ldots, x_{n_f}\}$ selected randomly, the algorithm searches for its two nearest neighbors: one from the same category (called the nearest hit) and the other from a different category (called the nearest miss). Once the nearest hit $x^h = \{x^h_1, x^h_2, \ldots, x^h_{n_f}\}$ and the nearest miss $x^m = \{x^m_1, x^m_2, \ldots, x^m_{n_f}\}$ have been found, we update the weight value for each feature as follows:

$$w_i = w_i + (x_i - x^m_i)^2 - (x_i - x^h_i)^2. \tag{3}$$

After that, a new feature vector is selected randomly and the process of updating the weights continues. Finally, the features with the highest weights are selected.

Although the original algorithm is supposed to be used only for binary classification, it can be easily generalized to multi-class problems [KSRS97]. The algorithm could also employ the kernel trick and, therefore, select features in the kernel space [CSS$^+$07]. Such feature selection method is well-suited for support vector machines [NT10, WRG$^+$10], because in the space of selected features a hyperplane separating individuals which belong to different categories can be found easier and more accurately. In [PVIII], RELIEF is used to reduce dimensionality of feature vectors for the problem of malware detection.

2.1.3.2 Genetic Algorithm

Genetic algorithm (GA) is stochastic algorithm in which the principles of organic evolution are used as rules in optimization. Genetic algorithm starts with an initial set of feasible solutions called population. Applied to the problem of feature selection, each individual in this population can be a binary vector in which one is placed in the i-th position if the i-th feature is selected, and zeros correspond to non-selected features.

GA tends to an optimal solution, using processes similar to evolution: crossover and recombination. Recombination, or mutation, is used to maintain genetic diversity from one generation of a population to the next. Mutation alters one or more values in an individual from its initial state, which potentially allows us to obtain better solutions:

$$\begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix} \rightarrow \begin{bmatrix}1 & 0 & 1 & 0\end{bmatrix} \tag{4}$$

A common method of implementing the mutation operator involves generating a random variable for each value in an individual. This random variable tells whether or not a particular value will be modified.

Crossover combines two solutions (parents) to produce a new solution (offspring). The idea behind crossover is that the new solution may be better than any of the parents if it takes the best characteristics from each of the parents. A simple crossover operator, called one-class crossover, randomly selects a point within a solution and then interchanges the two parent solutions at this point to produce two new offspring:

$$\begin{bmatrix} 1 & 0 & \mid & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \mid & 0 & 1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

There are many ways to implement a crossover: from the simple single-point crossover, described above, to more complicated crossovers [MCW04, ZC09].

Recombination and crossover contribute new solutions to the population during each iteration, i.e. generation, of the algorithm. After that, several solutions are selected for the next generation in such a way that better solutions have a higher probability to be selected for the new population. The quality of a solution is measured with the help of a fitness function. This function always depends on the problem posed. For the problem of feature selection, the fitness function can be defined as the cross-validation accuracy, which is discussed later in this chapter. Genetic algorithms where the best individuals survive with the probability of one are known as elitist genetic algorithms or genetic algorithms with elitism. Elitism guarantees the survival of the best element of the population, which guarantees that at least the fitness of the population measured as the fitness of the best individual does not decrease after the next iteration [SL06]. The algorithm stops when some stopping criterion is fulfilled, e.g. maximal number of generations has been reached, maximal number of function evaluations has been made, or some other similar criterion.

Genetic algorithm can be applied to an optimization problem when specialized techniques are not available or standard methods fail to give satisfactory answers [Her99, MGSK88]. GA is also used to automatically determine the relative importance of many different features and to select a good subset of features available to the system [PG93, SKLM05]. In articles [PI, PII], genetic algorithm is applied to solve a nonlinear, non-convex and non-separable integer programming problem, whereas study [PVII] uses genetic algorithm to select the most essential features and, therefore, to escape from high dimensionality in a classification task.

2.1.4 Dimensionality reduction

Dimensionality reduction is a process where new features are extracted from the input data to map this data to a space of fewer dimensions while the information content of the data remains the same. The feature selection process can be considered as a simple case of dimensionality reduction with a linear transformation function. Dimensionality reduction can be used to produce a compact low-dimensional encoding of a given high-dimensional dataset or simplify, reduce, and clean the data for subsequent supervised training.

Dimensionality reduction methods can be divided into two groups: supervised and unsupervised. As a rule, unsupervised dimensionality reduction techniques try to save as much information presented in the dataset as possible. Supervised algorithms are applied for data samples which are labeled as belonging to one of several predefined categories or groups. Usually supervised techniques utilize this label information to project data onto low-dimensional space in such a way that data samples belonging to different categories are separated.

Principal components analysis (PCA) is one of the most popular techniques for unsupervised dimensionality reduction [Jol86]. Classic PCA is an orthogonal linear transformation that maps the data to a new coordinate system where the axis directions contain maximal variance. These axes are ordered in such a way that the greatest variance by any projection of the data lies on the first coordinate, called the first principal component, the second greatest variance is on the second coordinate, and so on. This mapping is performed by analyzing eigenvectors of the covariance matrix calculated for extracted feature vectors. PCA can be extended by using a kernel function which allows us to perform a nonlinear data transformation, and, therefore, to model nonlinear variabilities in high-dimensional data [MMR$^+$01].

Locally linear embedding (LLE) is another unsupervised approach which address the problem of nonlinear dimensionality reduction. LLE identifies the neighbors of each data point, computes the weights that best linearly reconstruct each point from its neighbors, and, finally, finds the low-dimensional embedding vector which is best reconstructed by its weights [CY06]. Thus, LLE computes low-dimensional, neighborhood preserving embedding of high-dimensional data.

Laplacian Eigenmaps Method (LEM) is an approach closely related to LLE. It starts by constructing a weighted graph which includes a set of edges connecting neighboring points. Weights of this graph are used to define a Laplacian matrix and embedding vectors are calculated as eigenvectors of this matrix [BN03].

Multidimensional Scaling (MDS) offers an alternative perspective on unsupervised dimensionality reduction. It tries to map the original high-dimensional space to a lower dimensional space in such a way that distance between each pair of points is preserved. MDS calculates pairwise distances between data points in an original high-dimensional space and attempts to find points in new low-dimensional space so that they minimize the sum of differences between the corresponding distances in original and new spaces. Nowadays, there are many different versions of MDS proposed for different problems related to dimensionality reductions [BG05].

Isomap is a nonlinear generalization of classical MDS. It performs MDS, not in the input space but in the geodesic space of the nonlinear data manifold. The geodesic distances represent the shortest paths along the curved surface of the manifold measured as if the surface were flat. This can be approximated by a sequence of short steps between neighboring sample points. Isomap applies MDS to the geodesic rather than straight line distances to find a low-dimensional mapping that preserves these pairwise distances [Yan02].

Stochastic Neighbor Embedding (SNE) tries to embed the original high-dimensional data to a low-dimensional space so as to preserve neighborhood identity as well as possible [HR]. This is achieved by minimizing a cost function which is the sum of Kullback-Leibler divergences between high-dimensional and low-dimensional distributions over neighbors for each data sample. This cost is high if widely separated points in the low dimension are used to represent nearby points in the high dimension, and it is small when using nearby low-dimensional points to represent widely separated high-dimensional points. Thus, the SNE cost function focuses on retaining the local structure of the data in the new low-dimensional space.

Stochastic Proximity Embedding (SPE) generates low-dimensional Euclidean embeddings that best preserve the similarities between data samples supplied in the form of proximities [Agr03]. The SPE algorithm repeatedly selects pairs of samples at random and adjusts their coordinates so that their distances in the new low-dimensional space match more closely their respective proximities. These adjustments are controlled with the help of a learning rate parameter, which decreases during the simulation to avoid oscillatory behavior.

Semi-definite Embedding (SDE) is a variation of kernel PCA, in which the kernel matrix is learned from the data. It creates a graph in which each input is connected with its nearest input vectors and all nearest neighbors are connected with each other. This neighborhood graph is then analyzed with the help of semi-definite programming, which aims to find an inner product matrix that maximizes the pairwise distances between any two inputs that are not connected in the neighborhood graph while preserving the nearest neighbors' distances. A low-dimensional embedding is finally obtained by application of MDS on the learned inner product matrix [WS04].

Diffusion maps is a nonlinear geometric method of unsupervised dimensionality reduction. It preserves the diffusion distance between probability distributions centered at data points in the original space as Euclidean distance in the low-dimensional space. Coordinates in the new space are computed from the eigenvectors and eigenvalues of a diffusion operator on the input data. Diffusion maps focuses on discovering the underlying manifold from which the data has been sampled. By integrating local similarities at different scales, diffusion maps gives a global description of the dataset [CLL$^+$05].

One well-known method for supervised dimensionality reduction is based on the analysis of Fisher's linear discriminant [Fis38]. This algorithm finds a linear projection of the original high-dimensional data samples onto low-dimensional space such that the ratio of between-groups variability to within-groups variability is maximized. This problem is solved by finding eigenvectors and eigenvalues of the multiplication of the reverse within-classes scatter matrix and the between-classes scatter matrix. A kernel trick can also be applied to extend Fisher's discriminant analysis for data classes with nonlinear boundaries [MRW$^+$99].

Neighborhood Components Analysis (NCA) is a supervised dimensionality reduction method that learns a distance metric by finding a linear transformation of input data such that the average leave-one-out classification performance is maximized in the new low-dimensional space [GRHS04]. Leave-one-out classification predicts the class label of a single data point based on its nearest neighbors with a given distance metric. The transformation matrix can be found with the help of an iterative solver such as gradient descent.

Large Margin Nearest Neighbors (LMNN) is an supervised dimensionality reduction algorithm that tries to improve the classification accuracy of the k-nearest neighbors rule [WS09]. LMNN attempts to learn a linear transformation of the input space such that training inputs with different labels are widely separated and each training input shares the same label as its k-nearest neighbors. Extensions of LMNN include the use of multiple metrics instead of a single global metric, and the application of kernels for LMNN classification [WS09].

As described in the previous sections, genetic algorithm and RELIEF can also be considered as supervised methods of dimensionality reduction.

2.2 Data analysis and patterns extraction

After raw data have been preprocessed and the training set has been obtained, a data mining algorithm can be employed to build a model which accurately describes the structure of the data and allows one to extract required information from the dataset. The built model is supposed not only to extract knowledge patterns from the given data but also to be applicable to describe another dataset of similar kind. This thesis focuses on the three most popular classes of problems in the data analysis: classification, clustering and anomaly detection.

All these tasks rely on machine-learning algorithms that can be classified into three different types: supervised learning, unsupervised learning and semi-supervised learning. Supervised machine-learning algorithms require the training dataset to be properly labeled [Kot07], whereas unsupervised algorithms try to discover the structure of the data inside the training dataset and do not require the data be labeled [KP04]. Finally, semi-supervised machine-learning algorithms use a mixture of both labeled and unlabeled data to build models and therefore improve the accuracy of unsupervised methods [CSZ10].

2.2.1 Classification

In the problem of classification, each item in a set of data is classified into one of predefined set of categories or groups. The data analysis task in a classification problem is to find a function that assigns samples in a data collection to target classes. The goal of classification is to accurately predict the target class for each case in the data [KS13]. An example would be classifying network traffic as FTP, P2P, multimedia, etc. based on statistics about packet length, inter-packet timings and information derived from the transport protocol [LK12]. Classification has also many applications in customer segmentation [RS11], business modeling [KLS09], marketing [GX09], credit analysis [HH09], and biomedical modeling [RP05].

In the problem of classification, categories are discrete and do not imply order. The simplest type of classification problem is binary classification, in which the target class has only two possible values. Correspondingly, in a problem of multi-class classification the number of categories can be greater than two.

During the process of building a model, a classification algorithm finds relationships between the values of the extracted features and the values of the target category. Different classification algorithms use different techniques for finding these relationships. After the relationships have been found, they are summarized in a model, which can then be applied to a different dataset in which the class assignments are unknown [KS13]. As a rule, classification is based on supervised machine learning algorithms.

In this section, such well-known classification methods as multilayer perceptron and support vector machines are described in more detail. Other widely-used algorithms include Decision Trees [BFOS84], Bayesian Networks [FGG97], k-Nearest Neighbors [CD07], etc.

2.2.1.1 Multilayer Perceptron

Multilayer Perceptron (MLP) is one of the best-known supervised Artificial Neural Network (ANN) models [Hay98]. Like any ANN, a multilayer perceptron maps sets of input data onto a set of appropriate output. A regular MLP consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one, as shown in Figure 2. The number of layers is equal to three or more: an input layer, an output layer and one or more hidden layers. For a classification problem, a feature vector is supplied to the input layer, and the vector obtained on the output layer defines the class which this feature vector belongs to. For example, it can be implemented in such a way that the length of the output layer is equal to the number of different classes and the class corresponds to the index of the maximum value in the output vector.

Except for the nodes of the input layer, each node has a transfer function ϕ which uses data from neurones of the previous layer as variables and transfers the function value to the next layer. Using this function, the value in the i-th node of the l-th layer s_i^l can be calculated as follows:

$$s_i^l = \phi(\sum_{j=1}^{n_{l-1}} w_{ij}^l s_j^{l-1} + b_i^l), \tag{6}$$

where n_{l-1} is the number of nodes on the l-th layer, w_{ij}^l is the weight of the connection between the i-th node of the l-th layer and the j-th node of the previous $l-1$ layer, s_j^{l-1} is the output from the j-th node of the $l-1$ layer and b_i^l is the i-th value of the bias vector on the l-th layer.

As a rule, for training the network, MLP utilizes a supervised learning technique called back-propagation. This technique implies changing connection weights after processing each feature vector x based on the amount of error E_x in the output $y^o(x) = (y_1^o(x), \ldots, y_n^o(x))$ compared to the expected result $y^e(x)$. For

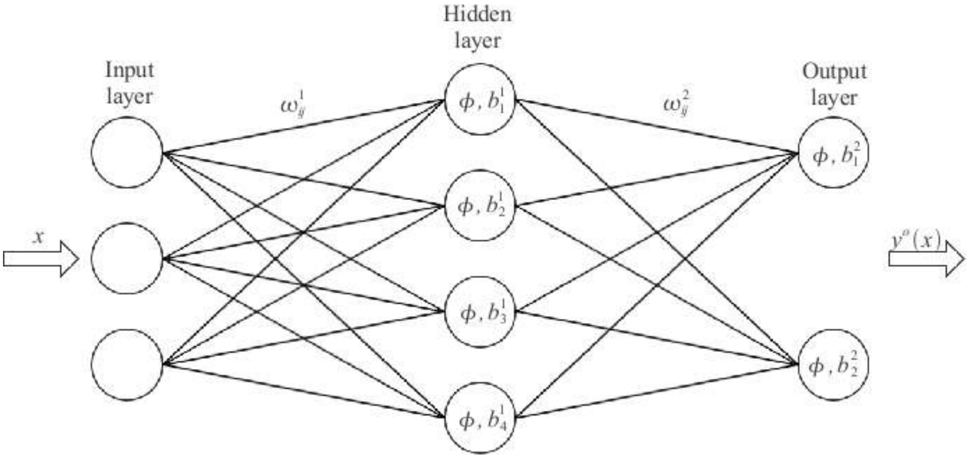


FIGURE 2 An example of multilayer perceptron

example, E_x can be considered as the mean square error:

$$E_x = \frac{1}{2}\sum_{i=1}^{n}(y_i^e(x) - y_i^o(x))^2. \tag{7}$$

Using the gradient descent algorithm, changes in each weight Δw_{ij}^l and bias Δb_i^l can be found as follows:

$$\begin{aligned} \Delta w_{ij}^l &= -\eta \frac{\partial E_x}{\partial w_{ij}^l}, \\ \Delta b_i^l &= -\eta \frac{\partial E_x}{\partial b_i^l}, \end{aligned} \tag{8}$$

where η is a parameter defining the learning rate and partial derivatives $\frac{\partial E_x}{\partial w_{ij}^l}$ and $\frac{\partial E_x}{\partial b_i^l}$ are calculated as follows:

$$\begin{aligned} \frac{\partial E_x}{\partial w_{ij}^l} &= s_j^l \delta_i^{l+1}, \\ \frac{\partial E_x}{\partial b_i^l} &= \delta_i^{l+1}. \end{aligned} \tag{9}$$

Here δ_i^{l+1} measures how much the i-th node of the l-th layer is responsible for an error in the output, and it can be obtained as

$$\begin{aligned} \delta_j^l &= -(y_j^e(x) - y_j^o(x))\phi', \text{ for the output layer,} \\ \delta_j^l &= \left(\sum_{i=1}^{n_{l+1}} w_{ij}^l \delta_i^{l+1}\right)\phi', \text{ for input and hidden layers.} \end{aligned} \tag{10}$$

Besides the backpropagation technique, MLP can use more advanced techniques such as Levenberg-Marquardt [Kis04] or quasi-Newton algorithms [SH95]. There are also several methods, in addition to the gradient descent algorithm, to adapt a multilayer perceptron. These methods include the steepest descent, the conjugate gradient method, the delta-bar-delta rule, SuperSAB [Rie94]. Regular MLP and its modifications are applied to various problems in diverse fields: from face recognition [HH89] to public key cryptography [YS02]. In study [PV], MLP is applied to a classification problem to predict the next user location.

2.2.1.2 Support Vector Machine

Support Vector Machines (SVMs) are supervised learning models that mostly are used for the problem of classification. As a rule, during the training an SVM takes a set of input points, each of which is marked as belonging to one of two categories, and builds a model representing the input points in such way that the points of different categories are divided by a clear gap that is as wide as possible. Thereafter, a new data point is mapped into the same space and predicted to belong to a category based on which side of the gap it falls on. Over the last decade, SVMs have been applied for many classification problems [BL02] because of their flexibility and computational efficiency.

SVM models can efficiently perform linear and nonlinear classification by mapping input vectors into high-dimensional feature spaces. By using a hyperplane in such a way that the distance from this hyperplane to the nearest data point on each side is maximized, a classic binary linear SVM model separates data samples belonging to two different categories. If such a hyperplane does not exist, a hyperplane that splits input points as cleanly as possible is chosen by the algorithm. In this case, a penalty function which measures the degree of misclassification of the data points is introduced for mislabeled points. Subsequently, the model is built to maximize the distance from the separating hyperplane to the nearest data point on each side, taking into account the penalties caused by mislabeled data points. In case when data classes have nonlinear boundaries, the kernel trick [ABR64] can be used to separate points by a hyperplane in a transformed feature space.

A regular SVM model classifies data belonging to two different categories (see Figure 3). Let us consider a classification problem where n samples belong to two categories: if sample x_i belongs to the first category, then it has label $y(x_i) = 1$, and otherwise $y(x_i) = -1$. In this case, the hyperplane (w, b) for the SVM model can be found after solving the following optimization problem:

$$\begin{aligned} &\min_{w,b,\xi_i} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i, \\ &\text{subject to} \begin{cases} y(x_i)(w^T \phi(x_i) + b) \geq 1 - \xi_i, \\ \xi_i > 0, \quad \forall i = \{1, \ldots, n\}, \end{cases} \end{aligned} \tag{11}$$

where ξ_i is the slack variable which measures the degree of misclassification of x_i, C is the penalty parameter and function $\phi(x)$ maps x to a higher dimensional

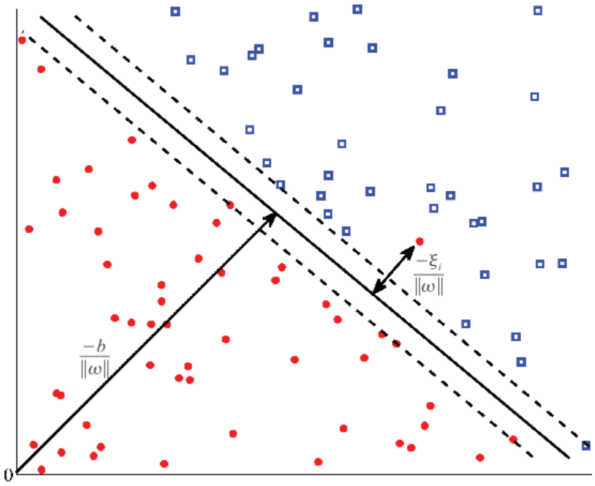


FIGURE 3 An example of binary support vector machine

space. Once optimal hyperplane (w, b) is found, a new sample x is classified as follows:

$$y(x) = \begin{cases} 1, \text{ if } s(x) \geq 0, \\ -1, \text{ if } s(x) < 0, \end{cases} \tag{12}$$

where function $s(x)$ is calculated as

$$s(x) = w^T \phi(x) + b. \tag{13}$$

If the feature vectors belong to more than two different classes, the classification is carried out based on the application of several binary SVM classifiers. One well-known strategy to deal with this is to build $(N_y)(N_y - 1)/2$ binary classifiers, where N_y is the number of different categories in the training set. Each such classifier is trained on data belonging to two different categories y_i and y_j and returns the corresponding function $s_{ij}(x)$, where $i, j \in \{1, \dots, N_y\}$ and $i \neq j$. A new sample x goes through all functions $s_{ij}(x)$, and it is defined either belonging to the i-th ($s_{ij}(x) \geq 0$) or the j-th ($s_{ij}(x) < 0$) category. Finally, the category of x is defined as the one which collects the most votes [Fri96]. Another solution is to use fuzzy multi-class SVM [AI02]. Based on fuzzy SVM, a vector x is classified as belonging to category $y(x)$ as follows:

$$\begin{aligned} & y(x) = argmax_{i \in \{1,\dots,N_y\}} m_i(x), \\ & \text{where } m_i(x) = \min_{j \in \{1,\dots,N_y\},\, j \neq i} (1, s_{ij}(x)). \end{aligned} \tag{14}$$

Multi-class SVM based on major votes is applied to the problem of predicting next user location in [PV]. In article [PVII], the problem of binary SVM classifiers combination is considered as a decision-making task, and it is solved with the help of game theory methods.

2.2.2 Clustering

Clustering is a division of data into groups of objects without knowing the structure of the dataset. Each such group is called a cluster and consists of objects that are in some way similar between themselves and dissimilar to objects of other groups [Ber02]. From a machine learning point of view, clustering corresponds to the unsupervised discovery of hidden patterns presented in the dataset to represent a data structure. Clustering plays a prominent role in the analysis of data in such areas as text mining [DHCW10], web analysis [YD11], marketing [TSD09], medical diagnostics [Alb03], and many others.

There are many different clustering algorithms which can be categorized based on the notation of a cluster. The most popular categories include hierarchical clustering algorithms [RVC12], centroid-based clustering algorithms [GT12] and density-based clustering algorithms [LP14]. Hierarchical clustering starts with a set of singleton clusters or a single cluster of all feature vectors and proceeds iteratively with merging or splitting of the most appropriate clusters until the stopping criterion is reached. In centroid-based clustering, clusters are represented by a central object, which is not necessarily a member of the dataset. In density-based clustering, clusters are defined as areas of higher density, while objects in sparse areas are considered as noise. In this thesis, a hierarchical single-linkage clustering algorithm and centroid-based clustering technique k-means are described in more details. A density-based clustering algorithm DBSCAN is presented in the next section as an algorithm of anomaly detection.

2.2.2.1 Single Linkage Clustering

Single-linkage clustering algorithm belongs to a class of agglomerative hierarchical clustering methods. In the beginning of the algorithm, each feature vector in the training dataset forms a cluster, i.e. the number of clusters is equal to the number of feature vectors, and every cluster consists of only one element. During the algorithm iterations, these clusters are sequentially combined into larger clusters.

The single-linkage clustering algorithm requires that the number of clusters is defined beforehand. At each iteration, the algorithm combines those two clusters which are the least distant from each other. The distance $d(C_i, C_j)$ between two clusters C_i and C_j is defined as the minimal distance between two vectors of these clusters, such that one vector is taken from each cluster:

$$d(C_i, C_j) = \min_{x \in C_i, y \in C_j} (d(x, y)). \tag{15}$$

The algorithm stops when the required number of clusters is formed. An example of the algorithm application with the predefined number of clusters equal to two is shown in Figure 4.

Study [Sib73] proposes a more advanced version of the single-linkage clustering algorithm, improving both compactness of storage and speed of operation. In [PVI], single-linkage clustering algorithm is used to classify n-grams extracted

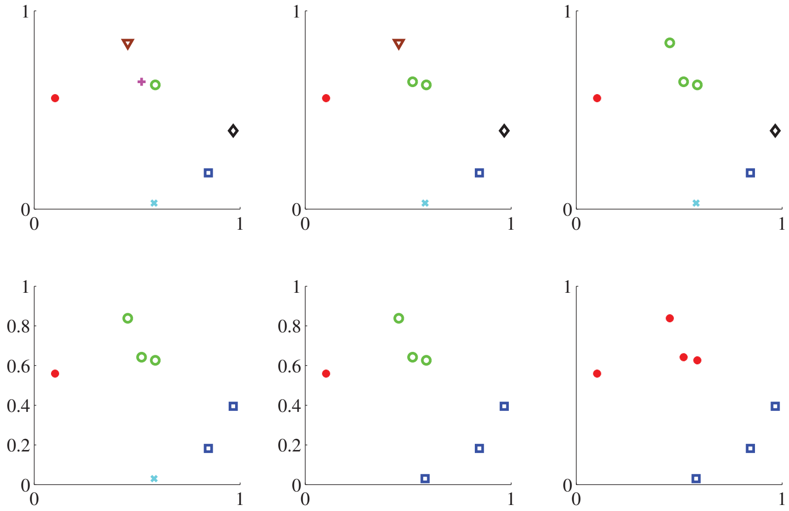


FIGURE 4 An example of the application of single linkage clustering algorithm with the number of clusters equal to two

from HTTP requests to define the most frequent ones. In [PIV], this algorithm is applied to find anomalous HTTP headers.

2.2.2.2 K-means and k-medoids

K-means is a unsupervised partitioning technique which classifies a dataset of objects into k clusters. As a rule, the number of clusters k is given beforehand. This algorithm tries to minimize the sum of Euclidean distances between each feature vector and the mean value of the cluster this vector belongs to.

The most common of these algorithms uses an iterative refinement technique [Llo06]. First, k means are initiated, e.g. by randomly choosing k feature vectors from the dataset. Let us denote these means as $m_1, \dots, m_k$. After that, each feature vector x is assigned to the cluster corresponding to the least distant mean. Thus, the i-th cluster C_i is formed as follows:

$$C_i = \left\{ x_j : ||x_j - m_i||^2 \leq ||x_j - m_l||^2 \; \forall l, \; 1 \leq l \leq k \right\}. \tag{16}$$

For recently built clusters, new means are calculated:

$$m_i^{new} = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j. \tag{17}$$

These two steps, the assignment of feature vectors to clusters and the calculation of new means, are repeated until there are no longer changes in clusters during the assignment step (see Figure 5).

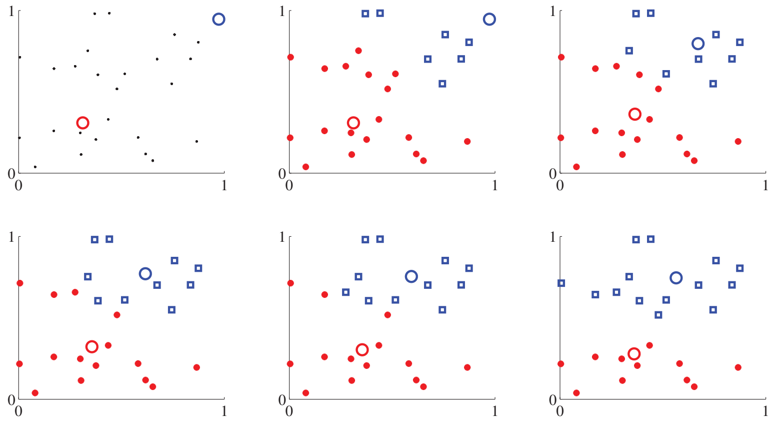


FIGURE 5 An example of the application of k-means with the number of clusters equal to two (circles denote mean values)

There is no guarantee that the k-means algorithm converges to the global optimum. Besides, the result strongly depends on the initial clusters [BMvL12]. There are several techniques for the initialization of means, e.g. random initialization [For65] or the minmax method [HS85]. K-means can also be applied in case the number of clusters is unknown [PSN05]. Study [PVI] applies k-means to classify n-grams extracted from HTTP requests.

The main drawback of the k-means algorithm is that the use of a distance function other than Euclidean distance may stop the algorithm from converging. K-medoids, which can deal with this problem, is a variation of k-means. The k-medoids algorithm tries to minimize the sum of dissimilarities between each feature vector and its medoid [PT10]. A medoid can be defined as an object of a cluster, whose average dissimilarity to all the feature vectors in the cluster is minimal. For all feature vectors in the dataset, the algorithm assigns each vector to the nearest cluster, depending upon the vectors distance from the cluster medoid. After every assignment of a data object to a particular cluster, a new medoid is calculated for this cluster. The k-medoids algorithm minimizes the sum of pairwise dissimilarities instead of a sum of squared Euclidean distances, and it is more robust to noise and outliers compared to k-means. In [PV], a supervised version of the algorithm based on k-medoids is used to solve the problem of the next user location prediction.

2.2.3 Anomaly Detection

Anomaly detection is the problem of finding patterns in data that do not conform to expected behavior [CBK09]. Anomaly detection finds extensive use in a wide variety of applications in different areas: from the analysis of medical time series [LKFH05] to intrusion detection in computer and network systems

[GTDVMFV09]. It also can be used as a part of preprocessing to remove anomalous data from a dataset. Removing anomalous feature vectors from the dataset may increase accuracy of a supervised machine learning technique applied subsequently [SM11].

As a rule, a problem of anomaly detection refers to the search for one of the following types of anomalies: point, contextual or collective. A point anomaly is the simplest anomaly type, and it can be defined as an individual data instance which significantly deviates from the rest of data. The majority of research nowadays focuses on the detection of anomalies of this type [QMG12, JS12]. A contextual anomaly is a data instance which is anomalous in a specific context. The problem of contextual anomaly detection is apparent where each data vector has features that can determine the context or neighborhood for that vector. Contextual anomalies have been most commonly explored in time-series data [GSCJ13] and spatial data [JYTK11]. A collective anomaly is a collection of related data instances which is anomalous with respect to the entire data set. The individual data instances in a collective anomaly may not be anomalies by themselves, but their occurrence together as a collection is anomalous. Collective anomalies can only be discovered in a data set in which data instances are somehow related. Collective anomalies can be found in sequence [KP12], graph [Ski07], or spatial data [SLZ01].

In this chapter, we consider two anomaly detection methods based on the cluster analysis performed by self-organizing maps and density-based spatial clustering of applications with noise. In addition, an anomaly detection method based on the description of a class of normal data with the help of support vectors is presented. Anomaly detection algorithms include also distance-based techniques such as k-nearest neighbors [ZS09] and local outlier factor [BKNS00].

2.2.3.1 Self-Organizing Map

Self-Organizing Map (SOM) is an unsupervised, competitive learning algorithm that allows us not only to compress high-dimensional data but also to create a network that stores information in such a way that any topological relationships within the dataset are maintained. Due to this, SOMs are widely applied for visualizing the low-dimensional views of high-dimensional data [SZWS09].

A regular SOM is formed from a grid of neurones, each of which is fully connected to the input layer. The i-th neuron of the SOM is associated with the n-dimensional prototype vector $p_i = [p_{i1}, p_{i2}, \dots, p_{in}]$, where n is equal to the dimension of the input vectors. Thus, the i-th neuron has two positions: one in the input space defined by the prototype vector p_i, and another in the output space on the map grid q_i.

During the training, the prototype vectors move so that they follow the probability density of the input data. At the beginning of training, the number of neurons, the dimensions of the map grid, the map lattice and shape, and the prototype vectors should be initialized. SOM is very robust with respect to the parameter initialization, but properly accomplished initialization allows the algorithm to converge faster to a good solution. At each training step t, one feature vector x_t from the input dataset is picked up randomly, and the distance between it and all the prototype vectors $p_i(t)$ is calculated. The prototype vector that is the least distant from input x_t is denoted as the Best Matching Unit (BMU) for x_t. Feature vector x_t is mapped to the location of the best matching unit, and the prototype vectors of the SOM are updated so that the vector of the BMU and its topological neighbors are moved closer to x_t in the input space:

$$p_i(t+1) = p_i(t) + \eta(t) N(d_{out}(q_{b(t)}(t), q_i(t)), r(t)) d_{in}(x(t), p_i(t)), \tag{18}$$

where $\eta(t)$ is the learning rate, $d^{in}(x(t), p_i(t)$ is the distance between $x(t)$ and the i-th prototype $p_i(t)$ in the input space, and $N(d_{out}(q_{b(t)}(t), q_i(t)), r(t))$ is the neighborhood function, which depends on neighborhood radius $r(t)$ and the distance between BMU $q_{b(t)}$ and i-th neuron in the output space $d_{out}(q_{b(t)}(t), q_i(t))$. An example of neighborhood function would be

$$N(d_{out}(q_{b(t)}(t), q_i(t)), r(t)) = e^{-\frac{d^2_{out}(q_{b(x(t))}(t), q_i(t)))}{2r^2(t)}}. \tag{19}$$

Thus, the effect of learning is proportional to the distance of a neuron from the current BMU. As a rule, the amount of learning is fading over distance, and at the edges of the BMUs neighborhood the learning process has almost no effect. Another important feature of the learning algorithm is that the area of the neighborhood shrinks over time. This is achieved with the help of the neighborhood radius function, which can be defined as follows:

$$r(t) = r_0 e^{-\frac{t}{\lambda}}, \tag{20}$$

where r_0 is the width of the map lattice and λ denotes a time constant. An example of the training process of SOM is shown in Figure 6.

Despite the fact that SOM is a clustering technique, it can be easily used in order to solve an anomaly detection problem [MD09, ROT03]. For example, it can be performed in such a way that when a new feature vector is mapped onto a trained SOM it is classified as normal if it is sufficiently close to its BMU; it is classified as anomalous if its distance from the BMU is more than a preset threshold [ROT03]. Another approach involves using the distance relationships and density structures in the training dataset [Ult05].

Despite SOM having been successfully employed for the analysis of high-dimensional data, the effectiveness of traditional SOM models is limited by the static nature of the model architecture. The size and dimensionality of the SOM model is fixed prior to the training process, and there is no systematic method for identifying an optimal configuration. Another disadvantage of the fixed grid in SOM is that traditional SOM cannot represent hierarchical relations that might be present in the data. Growing Hierarchical Self-Organizing Map (GHSOM) can resolve these limitations [DMR00]. GHSOM is a multi-layered hierarchical architecture which adapts its structure according to the input data. It is initialized with one SOM and grows in size until it achieves an improvement in the quality

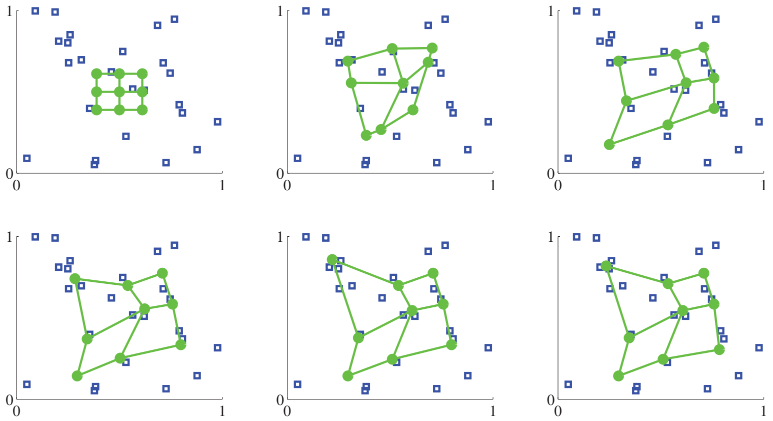


FIGURE 6 An example of the training of a self-organizing map in input space

of data representation. In addition, each node in this map can dynamically be expanded down the hierarchy by adding a new map at a lower layer, providing thus a further detailed representation of data.

As a rule, a GHSOM starts with the main node at zero layer and a 2 × 2 map at the first layer trained according to the SOM training algorithm. The main node represents the complete dataset and its prototype vector p^0 is calculated as the mean value of all feature vectors. This node controls the growth of the SOM at the first layer and the hierarchical growth of the whole GHSOM. The growth of each SOM in the GHSOM is controlled with the help of the quantization error

$$e_i = \sum_{x_j \in S_i} d_{in}(p_i, x_j), \tag{21}$$

where e_i is the quantization error for the i-th node, S_i is the set of feature vectors x_j projected to the i-th node, and p_i is the prototype vector of the i-th node. The quantization error E_m of SOM m is defined as

$$E_m = \frac{1}{|Q_m|} \sum_{i \in Q_m} e_i, \tag{22}$$

where Q_m is the subset of the m-th map nodes onto which data is mapped.

The key idea is that error E_m must be less than the certain fraction $\alpha_1 \in (0, 1)$ of error e_u of the corresponding parent unit u in the upper layer. If this condition is not satisfied, new neurons are added to the m-th map. For this purpose, the neuron j_e with the highest mean quantization error is found. After that, we select the neighboring neuron which is the most dissimilar to the i_e-th neuron:

$$j_d = \max_j (d_{in}(p_{j_e}, p_{j_d})), \text{ for } q_j \in N_{j_e}, \tag{23}$$

where N_{j_e} is the set of neighboring neurones of the j_e-th node. A new row or column of neurones is placed in between the j_e-th and the j_d-th neuron. The prototype vectors of the newly added neurones are initialized with the mean of their corresponding neighbors. The growing process is stopped when

$$E_m < \alpha_1 e_u. \tag{24}$$

Thus, the parameter α_1 controls the breadth of the maps.

After the SOM growth process is completed, every node of this SOM has to be checked for the fulfillment of the global stopping criterion:

$$e_i < \alpha_2 e_0, \tag{25}$$

where $\alpha_2 \in (0, 1)$ is the parameter which controls the hierarchical growth of GHSOM, and e_0 is the quantization error of the main node. Nodes which do not satisfy (25) represent a set of too diverse feature vectors. They are expanded to form a new map at a subsequent layer of the hierarchy. For this purpose, a new map of initially 2 × 2 nodes is created. This map's prototype vectors are initialized to mirror the orientation of the neighboring the units of its parent [CP02]. The newly added map is trained by using only the feature vectors mapped onto its parent. This new map grows, and the whole process is repeated for the subsequent layers until the global stopping criterion (25) is met by all neurones of the GHSOM.

Similar to regular self-organizing maps, GHSOMs are widely applied to various problems related to high-dimensional data analysis [RMD02]. In studies [PIII, PIV], a framework based on the use of GHSOMs is proposed for the detection of anomalous HTTP requests.

2.2.3.2 Density-based spatial clustering of applications with noise

Density-based spatial clustering of applications with noise (DBSCAN) is a powerful density-based clustering algorithm, which is often used for detecting outliers. It discovers clusters in the training dataset starting from the estimated density distribution of feature vectors [EKJX96]. All cluster-less vectors are classified as anomalies. DBSCAN requires two parameters: the size of neighborhood ε and the minimum number of points required to form a cluster N_{min}.

The algorithm starts with an arbitrary feature vector x that has not been checked. The number of feature vectors $N_\varepsilon(x)$ contained in the ε-neighborhood of x is found and compared to N_{min}:

$$\begin{cases} \text{If } N_\varepsilon(x) < N_{min}, & \text{then } x \text{ is labeled as noise,} \\ \text{If } N_\varepsilon(x) \geq N_{min}, & \text{then } x \text{ is a part of a cluster.} \end{cases} \tag{26}$$

Vectors marked as noise might later be discovered as a part of another vector ε-environment and hence be made a part of a cluster. If a vector is found to be a part of a cluster, its ε-neighborhood is also part of that cluster. After that, each point y contained in the ε-neighborhood is checked. If y is density-reached from x

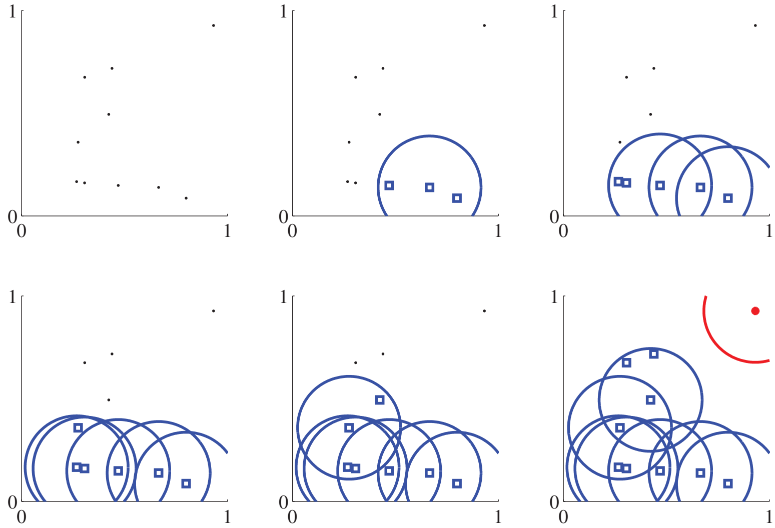


FIGURE 7   An example of the application of DBSCAN

with respect to ε and N_{min}, it is added to the cluster. Vector y is density-reachable from x with respect to ε and N_{min}, if there is a chain of points $x_1, x_2, \ldots, x_m$, where $x_1 = x$ and $x_m = y$, such that $\forall i \in \{1, 2, \ldots, m-1\}$ the two following conditions are satisfied:

$$\begin{cases} d(x_i, x_{i+1}) \leq \varepsilon, \\ N_\varepsilon(x_i) \geq N_{min}, \end{cases} \tag{27}$$

where $d(x_i, x_{i+1})$ is the distance between x_i and x_{i+1}. The cluster is built when all vectors density-reachable from x have been found. Then, a new unvisited vector is processed, leading to a discovery of a further cluster or noise. All points which remain cluster-less after the algorithm is finished are classified as anomalies. Figure 7 shows an example of the application of DBSCAN, with $N_{min} = 3$ and $\varepsilon = 0.25$ (radius of each circle).

In addition to discovered anomalies, DBSCAN can find arbitrarily-shaped clusters and does not require to know the number of clusters in the dataset a priori. DBSCAN requires just two parameters that should be optimally chosen. There are several studies devoted to this problem [Kim11, Smi12]. DBSCAN cannot be successfully applied to a dataset with large differences in densities of samples contained in it. However, some modifications of the algorithm can deal with this task [HYLZ09, XYYP08]. In [PVI], DBSCAN is used to cluster pseudodistances of HTTP queries from the normal set of n-grams and, therefore, find anomalous HTTP request messages.

2.2.3.3 Support Vector Data Description

By constructing a hypersphere which contains all data of one category, an SVDD finds a closed boundary around the data belonging to this category [TD04]. This sphere is characterized by center c and radius $R > 0$, as shown in Figure 8.

Let us assume that there are q data vectors $x_1, x_2, \dots, x_q$ which belong to one class, i.e. $y(x_i) = -1, \forall i \in \{1, \dots, q\}$. The center c of SVDD hypersphere (c, R) for this dataset can be defined as $c = \sum_{i=1}^{q} \alpha_i x_i$. Here $\alpha = (\alpha_1, \dots, \alpha_q)$ is the solution of the following optimization problem:

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{q} \alpha_i (\phi(x_i)^T \phi(x_i)) - \sum_{i=1}^{q} \sum_{j=1}^{q} \alpha_i \alpha_j \phi(x_i)^T \phi(x_j), \\ &\text{subject to} \begin{cases} \sum_{i=1}^{q} \alpha_i = 1, \\ 0 \leq \alpha_i \leq C, \forall i \in \{1, \dots, q\}, \end{cases} \end{aligned} \tag{28}$$

where function $\phi(x)$ maps x to a higher dimensional space and C is the penalty parameter which controls the trade-off between the hypersphere volume and classification errors. The radius R of the sphere (c, R) is calculated as follows:

$$R = \phi(x_k)^T \phi(x_k) - 2 \sum_{i:\alpha_i < C} \alpha_i \phi(x_i)^T \phi(x_k) + \sum_{i:\alpha_i < C} \sum_{j:\alpha_j < C} \alpha_i \alpha_j \phi(x_i)^T \phi(x_j), \tag{29}$$

where x_k is any vector from the dataset such as $\alpha_k < C$. Once optimal hypersphere (c, R) is found, a new sample x is classified as follows:

$$y(x) = \begin{cases} -1, \text{ if } d(x) \geq 0, \\ 1, \text{ if } d(x) < 0, \end{cases} \tag{30}$$

where function $d(x)$ is calculated as

$$d(x) = R^2 - (\phi(x)^T \phi(x) - 2 \sum_{i=1}^{q} \alpha_i (\phi(x)^T \phi(x_i)) + \sum_{i=1}^{q} \sum_{j=1}^{q} \alpha_i \alpha_j \phi(x_i)^T \phi(x_j)). \tag{31}$$

Here $y(x) = 1$ means that x is classified as an anomaly.

An SVDD can be easily applied for a problem of anomaly detection [WX11, YHGL10]. There are also modifications of SVDD which search for the optimal boundary hypersphere when "positive" feature vectors are present in the training dataset [TD04]. These vectors can be incorporated in the training to improve the description by finding a boundary hypersphere more accurately. If a dataset contains several classes of feature vectors, SVDD can be improved to give each class in the dataset a hyper-spherically shaped boundary [HCY10]. In [PVIII], SVDDs are used to build a benign software behavior model.

2.3 Algorithm performance evaluation

Not every pattern found by a data mining algorithm in a training set is necessarily valid for a more general dataset. For this reason, once a model which de-

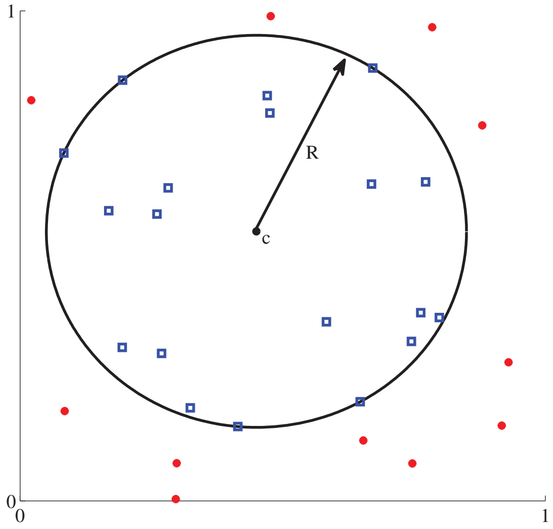


FIGURE 8 An example of support vector data description

scribes the structure of the input data has been built, this model is supposed to be evaluated to verify that it can describe a wider dataset. If the data used for training a data mining model is used to estimate the performance of the model, a non-realistic and over-optimistic prediction can be generated. For this reason, the evaluation uses, as a rule, a test set of data on which the data mining algorithm was not trained. The learned patterns are applied to this testing dataset, and the result is compared to the desired output.

2.3.1 Performance Metrics

Probably the most important metric for a data mining problem is the prediction accuracy, which refers to the ability of a model to correctly predict the class label of new or previously unseen data. Another property which should be taken into account is the computation costs involved in generating and using the model. In addition, a data mining model is supposed to show the ability to make correct predictions in the case of noisy data or data with missing values. Finally, a data mining model has to be reliable, i.e. it must generate the same type of predictions or find the same patterns regardless of the test data used.

The prediction accuracy is usually evaluated by using classification accuracy or misclassification error. Classification accuracy A is calculated as the ratio of the number of correctly classified testing samples N_{cc} to the number of testing samples N:

$$A = \frac{N_{cc}}{N}. \tag{32}$$

The misclassification error ε is the ratio of incorrectly classified testing samples $N_{uc} = N - N_{cc}$ to the number of testing samples N:

$$\varepsilon = \frac{N - N_{cc}}{N}. \tag{33}$$

In the case of a binary classification task, e.g. anomaly detection, there are several additional metrics that can be useful for a data mining algorithm evaluation. Let us label classes in a binary classification task as "positive" and "negative". In the problem of anomaly detection, "positive" class is usually corresponds to anomalies, whereas "negative" is used to denote normal instances. In this case, the following performance metrics can be introduced:

- True Positive Rate (TPR) is the ratio of the number of correctly classified "positive" samples to the total number of "positive" samples in the testing set,
- False Positive Rate (FPR) is the ratio of the number of "negative" samples classified as "positive" to the total number of "negative" samples in the testing set,
- True Negative Rate (TNR) is the ratio of the number of correctly classified "negative" samples to the total number of "negative" samples in the testing set,
- False Negative Rate (FNR) is the ratio of the number of "positive" samples classified as "negative" to the total number of "positive" samples in the testing set,
- Precision is the ratio of the number of correctly classified "positive" samples to the number of samples classified as "positive".

Another widely used approach to illustrate the performance of a binary classifier is a receiver operating characteristic (ROC) curve. It shows the dependence of TPR on FPR at various parameters used in the analyzed classification algorithm (Figure 9). ROC analysis allows us to select method parameters to reach the required value of TPR while minimizing FPR. The area under ROC curve (AUC) is usually used to represent the performance of a binary classification. For example, an AUC of 1.0 would mean that the tested algorithm could be used to perfectly classify samples. On the other hand, if AUC is equal to 0.5, it means that the algorithm is equivalent to random guessing.

2.3.2 Validation techniques

Validation is one the key points in any data mining process. Its main purpose is to predict how well the final model will work in the future or even whether it should be used at all. Validation helps to choose the most relevant technique for the given data mining problem and define optimal parameters for the chosen technique. When a huge number of data samples is available, the validation can be carried out in such a way that a large slice of data is used for training and another large portion can be used as test data. Such validation pattern is called

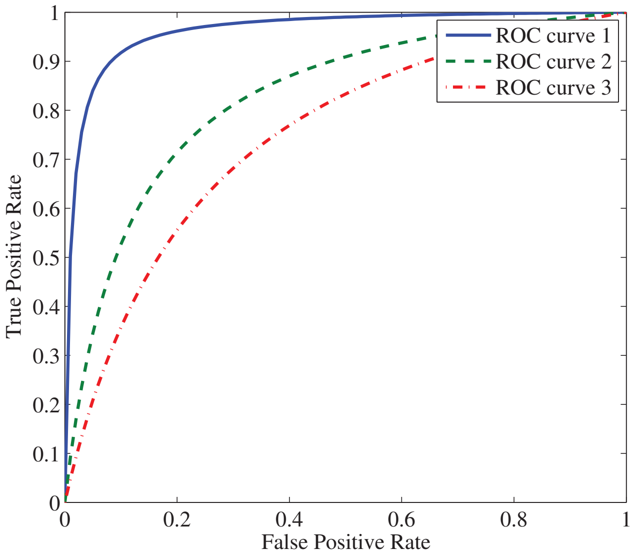


FIGURE 9 ROC curve examples

hold-out. On the other hand, when only a limited amount of data is available, more complex methods are needed to achieve an unbiased estimate of the model performance over unseen examples [SMJ02].

K-fold cross-validation approach divides the training set randomly into k equal-size subsets. One of these k subsets is retained as the validation data for testing the model and the remaining $k-1$ subsets are used as training data. The cross-validation process is then repeated k times, with each of the k subsets used exactly once as the validation data. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once. K-fold cross-validation is probably the most popular approach for validation of data mining algorithms [RTR+01, LZO04].

Leave-one-out is a validation scheme in which a data mining algorithm is trained on the full dataset, excluding only one sample against which it is then tested. Due to the use of training sets that are larger than those in the cross-validation technique, leave-one-out yields high accuracies. On the other hand, it can take a lot of time to validate an algorithm with this technique, especially in the case of large number of samples in the training dataset.

Bootstrap samples a subset of instances from the dataset with replacement. The number of samples in this subset is equal to the size of the original dataset. Next, a model is trained and evaluated with this subset calculating the error over this subset $e_{training}$ and over the remaining examples $e_{testing}$. The final error is estimated as $e = 0.632e_{training} + 0.368e_{testing}$. These steps are repeated to average the obtained error estimates. The number of repetitions is equal to the number of samples in the original training dataset [ET94].

3 RESEARCH CONTRIBUTION

In this chapter, the case studies and their results are presented. First, the problem of searching for optimal positions of relay stations in WiMAX multi-hop networks is considered. Next, the prediction of the next user location based on the analysis of spatial-temporal trajectories is explored. In addition, the contribution to the problem of detection of intrusive HTTP requests is shown. Finally, the problem of detection and classification of malware is discussed.

3.1 Optimal relay stations deployment

In WiMAX multihop networks, a mobile user can connect to a serving base station directly or via a relay station [PH09]. As a rule, a relay station has a lower cost compared to a base station and does not need to have any cable connection to the backhaul network. In addition, deploying relays is simpler and faster, because simpler equipment is used in it. Thus, the deployment of relay stations is one of the most promising means to improve the signal-to-noise ratio for users in poor radio conditions [CCLC09] and to extend the network service area [NHKH09].

Paper [PI] aims to position several relay stations in a WiMAX cell in such a way that the available bandwidth capacity is maximized. This problem is formulated as an integer programming problem. While such approach is not novel, existing research loses sight of the increasing interference caused by relay stations [CCLC09] or assumes that relays transmit a signal in different time frames and therefore do not interfere with each other [SWW08]. In [PI], a region containing several base stations is considered. The aim is to deploy several relay stations in this region, taking into account the interference increase caused by relay deployment. The proposed algorithm can also be used to find the optimal number of relays which should be placed in the region to maximize the available bandwidth and satisfy user throughput requirements.

To find optimal locations for relay stations, the considered region is divided into several small sectors distributed uniformly. For each sector, the available bandwidth capacity is defined as a function of relay stations' coordinates. These coordinates are used to determine which sectors are connected to base stations directly and which are connected through relay stations. The capacity is defined as a logarithmic function of a signal-to-noise ratio, where signal is equal to power of signal from a serving station and noise is the sum of power of signals from all other stations except the serving one [TV05]. After that, a nonlinear, non-convex and non-separable integer programming problem is formulated. For this problem, two variants of the objective are explored: the maximization of the sum of available bandwidth capacity values and the maximization of the minimal capacity value. The first variant allows the maximization of average available bandwidth per sector, while the second approach leads to a more uniform distribution of network resources. The optimization problem is solved with the help of an elitist genetic algorithm. Each individual in the algorithm is defined as a vector of relay station coordinates inside a cell.

The obtained solutions were simulated using network simulator ns-2 and its extension, Winse [SAM$^+$11]. Ns-2 is a packet-level network simulator that can model the access service networks to obtain true end-to-end simulation results. In the research community, it is widely used for evaluation of performance of different network topologies and protocols [GRS$^+$09, WSH$^+$10, AIJ11]. We consider the case of the deployment of 3 relay stations. The simulation results show that the proposed method of relay stations' optimal deployment is bandwidth-efficient and exhibits good fairness in multi-hop networks. Figure 10 presents the cumulative distribution function for the downlink connection throughput (CDF). The figure shows that when the total available bandwidth capacity is maximized, about 60% of the users have a lower throughput compared to when minimal available bandwidth is maximized.

Paper [PII] investigates the problem of cost-effective relay station deployment for extending network coverage area. This problem is usually considered as a search for relay positions so that bandwidth requirements would be satisfied for all subscribers in the fixed region considered as a network coverage extension [NHKH09]. In our research, we analyze a case where a provider has one base station already located in a new residential area and tries to attract new customers in this area by increasing the coverage area with the help of relay stations. To maximize the profits, the provider tries to find a balance between the increased income and costs caused by the relay deployment.

The provider's income is calculated based on the difference between the coverage area extended with relays and the area covered by the base station alone. The coverage area is defined in such a way that if the signal-to-noise ratio in the point is equal or greater than a given threshold then a point belongs to the area. Thus, the extended coverage area can be determined as a function of the number of relay stations deployed in the area and a set of these relay stations' coordinates. This function is found as the definite integral of the implicit function defined with the help of signal-to-noise value calculated for a given point of the coverage area. The provider's costs consist of one-time costs and recurring costs and can also be determined as the function of the number of relay stations deployed and the

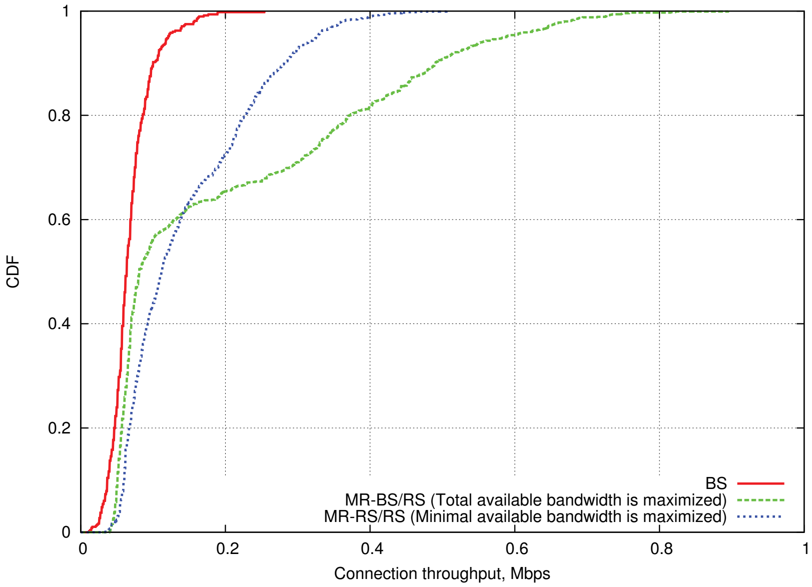
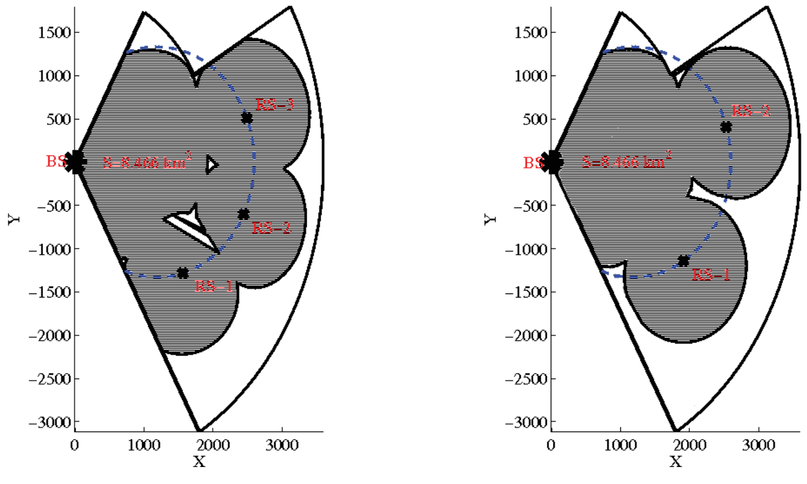


FIGURE 10 Cumulative distribution function for the downlink throughput in the case of three relay stations deployed in the cell

coordinates of their positions. Thus, we can formulate a maximization problem, in which the objective function is defined as the difference between the provider's income and costs as a function of the number of relays and their coordinates. Similarly to the previous study, this optimization problem is solved with the help of a genetic algorithm.

We considered an example of residential area where one base station is located. The numerical results for this region show that the coverage area starts shrinking after the deployment of a large number of relay stations. This is caused by the increase of the interference level in the considered area. After that, the problem of cost-effective relay deployment was solved for different values of costs of the relay stations. The coverage area is shown in Figure 11 as the shaded area. The provider's income and costs calculation proves that the provider's profit can be increased drastically by using the proposed mechanism.

Although the considered problems of the optimal deployment of relay stations cannot be considered as directly related to data mining applications, they are solved with the help of a genetic algorithm which is widely-used in data analysis. Genetic algorithm is often applied for the selection of the most essential features. Genetic programming is a machine-learning technique based on genetic algorithm, which is employed in problems of regression and classification [Kor07]. In addition, many machine-learning algorithms, e.g. support vector machines and multilayer perceptrons, are based on the solution of optimization problems. For this reason, the studies described above can be considered as a contribution to the field of data mining.



(a)

(b)

FIGURE 11 Relay stations optimal positions and coverage area for different values of one-time costs: (a) low one-time costs, (b) high one-time costs

3.2 Prediction of the next user location

Prediction of the next location of a mobile user is a challenging data mining task that can be applied for recognizing human behavior and forecasting the dynamics of crowds. There are several studies in which this problem is solved based on social behavior of users [CML11, GLJ$^+$11]. However, it is usually very difficult or even impossible to define a social interrelation between different users based on the analysis of the data gathered from their mobiles. Due to this fact, user-specific prediction techniques based on spatial trajectories and temporal patterns seem more reasonable to use [LSPS11, GTL12].

In study [PV], we apply a user-specific approach for the problem of a mobile user's next-location prediction based on temporal-spatial patterns. For this purpose, we analyze data collected from mobile phones of several non-related users over different periods of time. For each mobile user, spatial and temporal features are extracted from these data. Spatial features include only the current user location, while temporal features contain the day of the week and the time of the day when the user changes location and the amount of time spent in the current place before moving to the next location. Once a feature matrix has been constructed, the problem of the next-location prediction is reduced to a classification problem. This classification problem is supposed to be solved by taking into account constant changes in the user's habits.

In this study, classification methods such as k-medoids, support vector machine and multilayer perceptron are applied. The k-medoids algorithm is modified to classify objects in a supervised way. For each mobile user, several clusters

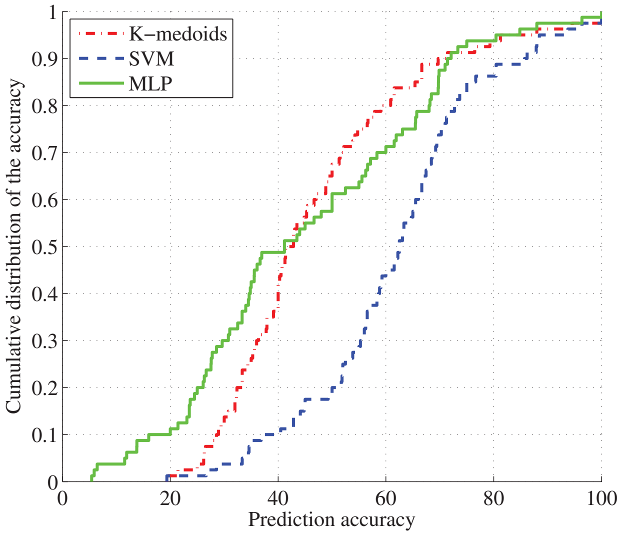


FIGURE 12 The cumulative distribution of the prediction accuracy obtained with help of k-medoids, SVM and MLP

are created and filled with feature vectors. The labels of these clusters are based on the possible locations of this user. To make the algorithm capable of adjusting to changes in the user's habits, only feature vectors with maximal score participate in the building of the clusters. This score value is calculated based on how frequently and how recently the user visits a location. Multi-class support vector machines with linear and Gaussian kernels are also employed for the problem. In order to solve the problem of possible changes in a user's behavior, SVM is trained using only a portion of the last temporal-spatial patterns contained in the training set. Finally, a multilayer perception with three layers of neurons is used. Since the present temporal-spatial patterns are more useful than the patterns from long time ago for predicting the user's moves in the future, the MLP model employs a forgetting function during the training phase. Optimal parameters for each classifier are found using the validation dataset which contains patterns from the last portion of the training set. For each user, the most accurate classifier is defined and the user's next place is predicted based on this best classifier.

The algorithm was tested with a set containing data collected from mobile phones of several non-related users. The cumulative distribution function of the prediction accuracy for all users is shown in Figure 12. As one can see, on average SVM outperforms other classifiers, and it is selected as the most accurate classifier for most users. However, for several users, either k-medoids or ANN is chosen as the best classifier. The proposed algorithm shows better results compared with its analogues, as can be seen from Table 1.

TABLE 1 Average accuracy of the algorithm for the next-location prediction based on k-medoids, SVM and MLP compared with analogues

Algorithm	Accuracy
Most frequently visited places	35.00 %
First order Markov chains	44.88 %
Hierarchical Pitman-Yor Prior Hour-Day Model	50.53 %
Gradient Boosted Decision Trees	57.63 %
Dynamical Bayesian Networks	60.07 %
Artificial Neural Networks	60.83 %
Algorithm based on k-medoids, SVM and MLP	62.83 %

3.3 Detection of anomalous HTTP requests

Computer networks and systems are vulnerable to different forms of intrusions. One of the most popular attack targets is web-servers and web-based applications. Usually, users of such applications request and send information using queries, which in HTTP traffic are strings containing a set of attributes. It is possible to manipulate these queries and create requests which can corrupt the server or collect confidential information [NTGG$^+$05]. One means to ensure the security of web-servers and web-based applications is the use of intrusion detection systems. There are two basic approaches for detecting intrusions from network data: misuse detection and anomaly detection. In the misuse detection approach, the IDS scans the computer system for predefined attack signatures. This approach is usually accurate, but cannot detect attacks for which it has not been programmed [Gol06]. The anomaly detection approach learns normal behavior patterns and classifies patterns that deviate from established norms as anomalies. Thus, systems which use anomaly detection are able to detect zero-day attacks [KV03]. However, the number of false alerts increases because not all anomalies are intrusions.

Paper [PIII] focuses on the analysis of dynamic HTTP requests, which are handled by the Web-applications of a web-server because it is difficult to inject code via static queries unless there are major deficiencies in the server itself. It is pointed out that most HTTP requests which are coming to the HTTP server are normal, i.e. the portion of intrusive requests is small. The study concentrates on that part of each dynamic request which consists of the path to the desired web resource and a query string which is used to pass parameters to the referenced resource. Based on the analysis of these parts, the method for the detection of anomalous HTTP requests is proposed.

First, the n-gram character model is applied to transform each HTTP request to the sequence of n-characters. Each such sequence is represented as a sequence of arrays each of which contains n decimal ASCII codes. A feature vector is built by counting the number of occurrences of each such array in the analyzed request. Feature vectors are separated into several feature matrices based on the web resource they are related to. Each feature matrix is used to train a growing hierarchical self-organizing map. After each GHSOM has been trained, anomalous neurons are found by calculating U^*-matrices for each SOM [Ult05]. Each such matrix represents a combination of distance relationships and density relationships and can give an appropriate clustering. Anomalous neurons correspond to high values of a U^*-matrix, since most requests are normal and intrusions are mapped to neurones which are located on cluster borders. A new HTTP request is classified as an intrusion if the distance between this request and its BMU prototype vector is greater than a threshold or if the BMU is an anomalous neuron. The study states that GHSOMs should be retrained after a certain period of time to be capable to classify new requests.

The proposed method was tested using logs acquired from a large real-life web service. These logs contain mostly normal traffic, but they also include intrusive HTTP requests. In the first simulation, a GHSOM was constructed for each unique web resource. The U^*-matrix for one of these GHSOMs is shown in Figure 13. Almost all real attacks were classified correctly as intrusions when 2-gram and 3-gram models were used. At the same time, the number of false alarms remained very low. However, since the construction of one GHSOM for each unique web resource is disadvantageous in terms of time and memory, the paper proposes to combine HTTP requests that have a similar structure to one group. After that, for each such group a GHSOM is constructed. The U^*-matrix for one of these new GHSOMs can be seen in Figure 14. In this case, the accuracy of the method remains very high.

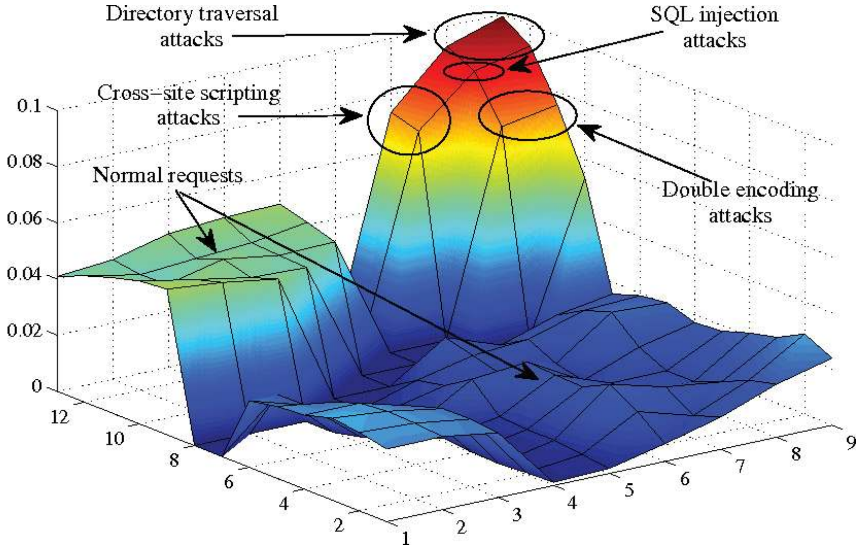


FIGURE 13 U^*-matrix of the GHSOM after the training stage (one web resource)

In paper [PIV], the method described in [PIII] is extended to allow one to find code injections in HTTP header fields. Such fields usually contain information about user agent, preferred response languages, connection type, and other

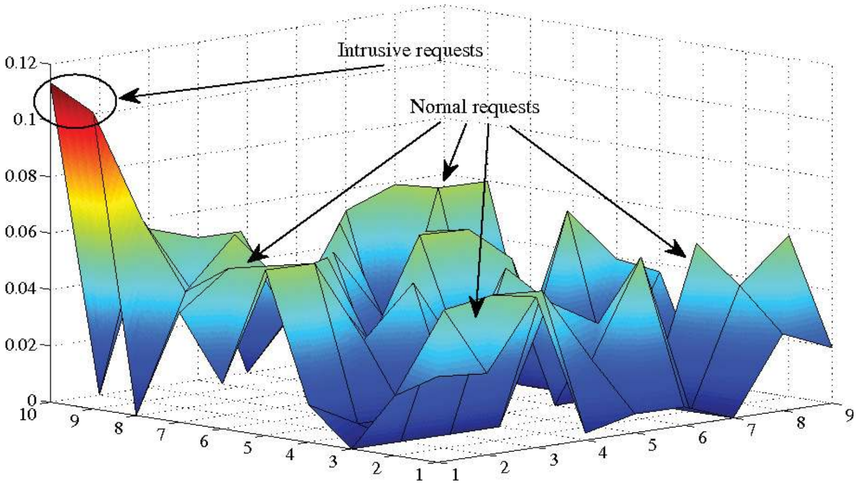


FIGURE 14 U^*-matrix of the GHSOM after the training stage (one group of web resources)

things. An attacker can inject malicious code to these fields to create attacks based on HTTP response splitting or malicious redirecting.

Different header types are analyzed separately. For each header field, its length and all non-alphanumeric symbols are extracted. Two distance functions are introduced based on the mean and variance of the lengths of header fields and the frequencies of non-alphanumeric symbols used in them [CG10]. All length and frequency values are classified into two clusters by using a single-linkage clustering algorithm. All anomalies are mapped to a small cluster and removed from the model. The remaining values are used to define normal values of headers. A new HTTP request is classified as anomalous if one of its header fields contains anomalous non-alphanumeric symbol or if its length is greater than a threshold value. As previously, almost all real attacks were correctly classified as intrusions, and the false positive rate was about zero.

Paper [PVI] describes an algorithm for processing HTTP queries which allows the analysis of all HTTP request messages at once without separating them by web resource. The algorithm proposed can be used to classify HTTP requests even in the case the set of requests free of attacks cannot be extracted. The technique is based on the extraction of features with the help of more advanced n-gram models and the analysis of feature vectors by simple clustering techniques.

The algorithm starts with the extraction of features for each HTTP query and header field. These features include all unique symbol n-grams found in a query, frequencies of their appearance and the length of each query. For each n-gram, the frequency of its usage in the training set is calculated. Based on these frequency values, all n-grams are divided into two groups by using k-means or the single-linkage clustering algorithm. The cluster with the biggest centroid value is considered as the one containing the most popular n-grams. N-grams belonging to the second cluster are denoted as abnormal. After that, a function that measures the "distance" between an HTTP request, and the set of popular n-grams is introduced. This function takes into account the frequency of appearance of abnormal n-grams extracted from this request, frequency of usage of those n-grams in the training set and the length of the request. This function is applied to all HTTP requests contained in the training set. The values of this function are then clustered by applying DBSCAN. All cluster-less values are removed from the model. A threshold is defined by using the remaining "distance" values. A new HTTP request is classified as an intrusion if the value of the "distance" function calculated for this request is greater than this threshold.

To test the method, HTTP logs of a large web service are used. Normal HTTP queries are generated based on requests to this service. In addition, several types of intrusive requests and header injections are added. The application of the proposed scheme showed that all HTTP requests to one web resource were classified correctly with the help of n-gram models when *n* is equal to two or three. The algorithm's performance results compared to several other methods are presented in Tables 2 and 3.

TABLE 2 Performance of the method for detecting intrusive HTTP queries based on advanced n-gram and DBSCAN compared to other techniques (one web resource)

Algorithm	True positive rate	False positive rate	Accuracy	Precision
K-nearest neighbor	59.31 %	0.06 %	97.93 %	97.99 %
N-gram + GHSOM	92.51 %	0.19 %	99.45 %	96.21 %
N-gram + Diffusion Maps	98.72 %	0 %	99.94 %	100 %
Advanced n-gram + DBSCAN	100 %	0 %	100 %	100 %

TABLE 3 Performance of the method for detecting intrusive HTTP queries based on advanced n-gram and DBSCAN compared to other techniques (several web resources)

Algorithm	True positive rate	False positive rate	Accuracy	Precision
K-nearest neighbor	55.51 %	2.05 %	97.79 %	59.12 %
N-gram + GHSOM	58.22 %	0.86 %	97.05 %	78.56 %
N-gram + Diffusion Maps	97.37 %	23.15 %	77.94 %	19.11 %
Advanced n-gram + DBSCAN	100 %	0 %	100 %	100 %

As one can see, all algorithms show good results when only one web resource is explored. When the number of resources grew, the accuracy of other methods decreased, but the accuracy of the algorithm proposed remained the same. Thus, the results proved that the method allows us to analyze all HTTP request messages without separating them by resource. However, this method is recommended to be used only in the case of simple HTTP requests that contain textual or numerical attributes.

3.4 Malware detection and classification

Today, malicious software, or malware, remains a significant threat to network security [MBM+12]. Trojan horses, spyware and Internet worms are designed by attackers to harm computers or networks to allow them to profit at the expense of ordinary users. The most popular commercial malware detection approach based on a search of signatures is not able to deal with constantly appearing new types of malware and variants of the existing ones until an instance of this malware has damaged several computer networks or systems. One method to handle this problem relies on the application of data mining algorithms [SWL08].

In study [PVII], a data-mining-based approach is applied to detect and classify malware among executable files. It is pointed out that a quite big set of properly labeled executable files is available. Most of these files are goodware. However, few files belong to different types of malware. The aim is to employ a supervised machine-learning technique to train a model which is able to detect these malware types in future. In addition, the detected malware is supposed to be classified so that to enable selection of a specific action to this malware depending on its type.

In the beginning of the algorithm, files of the training set are presented in the form of byte and opcode sequences. By using an n-gram word model, these sequences are transformed to feature vectors which express the frequency of appearance of every n-byte and n-opcode. To reduce time and computing resources when classifying these vectors, a genetic algorithm is employed to select the most essential features and, therefore, escape from the high dimensionality of the problem. This algorithm uses classification accuracy as the fitness function. A classification model is then built with the help of several binary support vector machines. For the feature vector extracted from a new executable file, values of functions defined by these binary classifiers are calculated. Based on these values, a zero-sum matrix game [Rom97] with strategies corresponding to different SVM classifiers is constructed. If this game has a saddle point in pure strategies, the label which corresponds to this point is assigned to the new file. Otherwise, the file is classified based on the saddle point in mixed strategies.

The method was tested using opcode and byte sequences extracted from real executable files, some of which were infected with malware. The proposed algorithm applied to byte sequences showed better results in terms of the true positive rate in case of the 2-gram model. On the other hand, when the 1-gram model is employed, opcode sequences obtain fewer false alarms. The performance of the algorithm compared with well-known classifying techniques applied along with several dimensionality reduction techniques is presented in Tables 4 and 5. The comparison results prove that binary support vector machines integrated with the game-theoretic approach and genetic algorithm outperform all other techniques in terms of classification accuracy.

TABLE 4 Malware detection and classification accuracy (in brackets) of the algorithm based on binary SVMs and zero-sum matrix games (ZSGSVM) compared to analogues (in percent) applied to opcode sequences

Algorithm	LDA	PCA+NCA	PCA+LMNN	RELIEF	GA
MLP	89.58 (85.76)	92.01 (89.93)	86.00 (82.25)	94.44 (87.15)	88.89 (85.42)
DT	93.06 (91.67)	89.24 (88.54)	81.94 (78.13)	96.88 (93.07)	93.40 (89.24)
KNN	94.79 (92.01)	94.79 (92.71)	93.06 (90.63)	97.22 (95.14)	94.10 (92.36)
SSDBSCAN	92.36 (92.01)	95.83 (92.36)	94.10 (90.63)	91.32 (90.63)	92.71 (92.01)
MVSVM	93.06 (87.15)	94.79 (93.06)	94.44 (93.06)	96.88 (94.44)	**97.57** (95.83)
FSVM	93.06 (87.15)	94.79 (93.06)	94.44 (93.06)	96.88 (94.44)	**97.57** (95.83)
ZSGSVM	93.06 (85.76)	94.79 (93.06)	94.44 (93.06)	96.88 (95.14)	**97.57** (**96.18**)

TABLE 5 Malware detection and classification accuracy (in brackets) of the algorithm based on binary SVMs and zero-sum matrix games (ZSGSVM) compared to analogues (in percent) applied to byte sequences

Algorithm	LDA	PCA+NCA	PCA+LMNN	RELIEF	GA
MLP	92.71 (86.46)	89.58 (85.76)	85.50 (83.50)	88.19 (84.38)	88.89 (87.50)
DT	95.14 (91.67)	91.32 (90.28)	86.46 (83.68)	96.18 (94.10)	94.44 (91.32)
KNN	95.49 (94.79)	95.14 (93.75)	95.83 (94.79)	96.53 (95.14)	96.53 (94.44)
SSDBSCAN	93.40 (89.93)	94.10 (91.67)	93.75 (91.67)	92.36 (91.32)	93.75 (91.67)
MVSVM	96.86 (95.14)	**98.26** (94.79)	89.93 (88.89)	95.49 (94.79)	**98.26** (95.83)
FSVM	96.86 (95.14)	**98.26** (94.79)	89.93 (88.89)	95.49 (94.79)	**98.26** (95.83)
ZSGSVM	96.86 (95.14)	**98.26** (95.49)	89.93 (89.24)	95.49 (94.79)	**98.26** (**96.18**)

However, the proposed method cannot be used to detect new malware types.

Paper [PVIII] focuses on the construction of a benign software behavior model by using a supervised anomaly detection approach. In this study, the training set consists of executable files, some of which are known as malicious executables or files already infected by malware. The aim is to build a model which would allow us to detect malware among new executable files even if types of these malware are not presented in the training set.

Similarly to the previous paper, each executable file is represented as a sequence of opcodes. An n-gram word model is applied to transform each such opcode sequence to an n-gram feature vector, which expresses the frequency of every n-opcode in the analyzed sequence. The most relevant features are selected with the help of RELIEF. After that, the benign software model is constructed by iteratively applying support vector machines. These are applied in such a way that separation of all feature vectors which correspond to goodware into several groups becomes possible. During this process a binary or fuzzy multi-class SVM is trained based on feature vectors corresponding to files belonging to these groups and to the group of feature vectors extracted from malware executables. Once all files have been classified, the label correction procedure starts. On this stage, feature vectors are regrouped to represent relationships between files in the training set more accurately. In addition, support vector data descriptions are employed for each group of feature vectors extracted from benign software files. A new executable file is classified based on either SVM or SVDD, depending on which of those classifiers works better on the files of the training set.

As previously, we tested the algorithm, using opcode sequences extracted from real executable files. The set of files was divided into a training set and a testing set in such a way that all the infected files in the testing set would belong to malware types not presented in the training set. First, the iterative SVM clustering and SVDD were applied along with different dimensionality reduction techniques to check the convergence of the algorithm. It is shown that the scheme takes only few iterations to converge and the resulting number of goodware groups strongly depends on the dimensionality reduction technique. For low values of false positive rate, the method shows the best results in terms of true positive rate when RELIEF is applied to find the most relevant features (see Figure 15). Finally, the proposed algorithm was compared with well-known supervised classifiers and outlier detection techniques. The comparison results are listed in Table 6. As one can notice, the algorithm based on iterative SVM clustering and SVDDs outperforms all other techniques in terms of the zero-day malware detection accuracy.

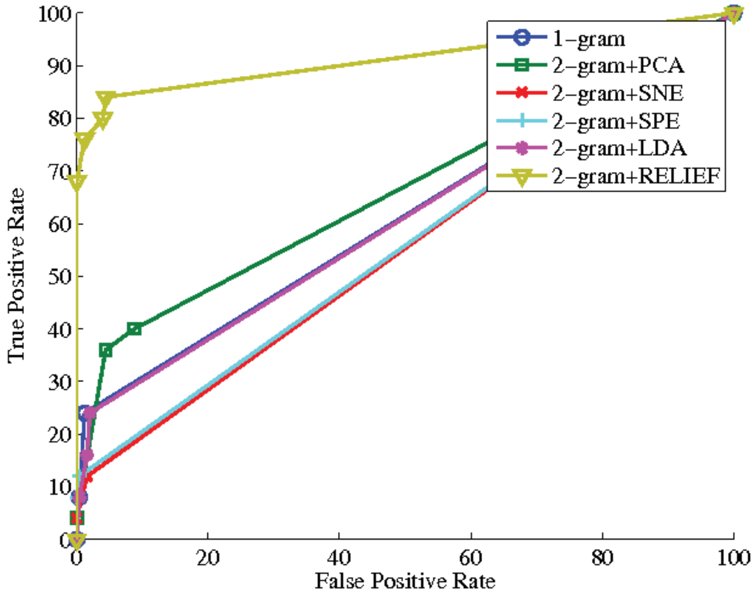


FIGURE 15 The dependence between false positive and true positive rates of the malware detection scheme based on iterative SVM clustering and SVDD for different n-gram models and dimensionality reduction techniques

TABLE 6 Zero-day malware detection accuracy (in percent) of the algorithm based on iterative SVM clustering (ISVMC) and SVDDs compared to analogues

Algorithm	1-gram	2-gram + PCA	2-gram + SNE	2-gram + SPE	2-gram + LDA	2-gram + RLF
MD	89.82	90.55	91.64	90.55	92.73	90.91
K-means	88.00	89.82	88.00	92.36	86.18	90.55
SOM	73.45	86.91	88.73	88.73	88.36	87.27
KNN	90.18	90.91	90.55	90.55	92.73	89.45
DBSCAN	90.91	90.91	91.64	90.91	91.64	90.91
LOF	90.18	90.55	92.36	92.36	91.27	74.55
SVM	91.64	91.27	91.27	90.91	90.55	92.36
OCSVM	89.82	90.91	90.18	90.18	91.27	84.36
ISVMC+SVDD	91.27	91.27	91.27	90.91	92.73	97.09

4 CONCLUSION

Nowadays, tasks related to the analysis of huge amounts of data are encountered in various areas of computer science. As a rule, the aim of such analysis is to extract knowledge patterns from these data for further use. This discovery process is called data mining. This dissertation describes the main phases of the data mining process and presents the application of different data mining algorithms to several problems encountered in mobile networks and network security.

This thesis can be divided into two main parts. The first part considers three main phases of the data mining process, which are data preprocessing, data analysis and validation of the result. These phases are further described in this dissertation. The most important steps of each phase are outlined and several implementation schemes are presented.

In the second part, several case studies devoted to different problems in the field of computer science are presented. Each of these studies employs one or more data mining techniques in order to solve a posed problem. Firstly, genetic algorithm is used to find optimal positions of relay stations in WiMAX multi-hop networks. Next, the prediction of the next-mobile-user location is carried out based on the application of several classifying methods to spatial-temporal trajectories of the user. Several clustering and anomaly detection techniques applied to detect intrusive HTTP requests are then presented. Finally, the data mining approach is applied for the detection and classification of malicious software.

In the next stage of our research, we are planning to apply the data mining approach to detect network intrusions based on flow-based traffic analysis. A flow-based intrusion detection approach uses information in packet headers, and, therefore, it has to handle a considerably lower amount of data than payload-based approaches. Thus, this approach can be employed in high-speed networks for the detection of certain types of attacks such as Denial of Service (DoS), scans, worms and botnets. In addition, we are going to consider the problem of the detection of malware designed for mobile devices. The popularity of smartphones and tablets has led to the spread of mobile malware. However, the accuracy of methods applied for their detection remains low.

YHTEENVETO (FINNISH SUMMARY)

Tämä väitöskirja, *Tiedonlouhinta-algoritmien käyttö matkaviestinverkoissa ja verkon turvallisuudessa*, keskittyy erilaisten tiedonlouhintamenetelmien ja -algoritmien soveltamiseen erilaisiin tietotekniikan ongelmiin. Tiedonlouhinta on aineistojen analysointiprosessi, jonka tarkoituksena on tietämysrakenteiden löytäminen ja mallin rakentaminen jatkokäyttöä varten perustuen näihin rakenteisiin. Tämä analysointiprosessi sisältää tietokantojen hallintaa, tietojen esikäsittelyä, mallin valintaa, vaativuuden huomiointia, löydettyjen rakenteiden jälkikäsittelyä, visualisointia ja päivittämistä. Nykyään erilaisia tiedonlouhintamenetelmiä ja algoritmeja käytetään liiketoiminnassa, päätöksenteossa, lääketieteessä, biologiassa ja ohjelmistotuotannossa. Tässä väitöskirjassa kuvataan tärkeimmät tiedonlouhintaprosessin vaiheet, käsitellään useita tiedonlouhinta-algoritmeja ja esitellään useita tapaustutkimuksia, joissa erilaisia tiedonlouhinta-algoritmeja käytetään useiden matkaviestinverkkojen ja verkon turvallisuuteen liittyvien ongelmien ratkaisemiseen.

Väitöskirja voidaan jakaa kahteen osaan. Ensimmäisessä osassa tarkastellaan tiedonlouhintaprosessin kolmea päävaihetta, jotka ovat tietojen esikäsittely, analysointi sekä tuloksen validointi. Esikäsittely on aineistojen analysoinnin alkuvaihe, joka voi sisältää tarvittavien tilastojen poimimista raakadatasta, tärkeimpien piirteiden valintaa, ulottuvuuksien vähentämistä, standardointia ja normalisointia. Tiedon analysointivaiheen aikana valitaan asianmukainen tiedonlouhinta-algoritmi, jotta voidaan rakentaa malli, joka kuvaa tiedon rakenteen tarkasti ja mahdollistaa tarvittavan tiedon poimimisen aineistosta. Viimeisessä tiedonlouhintaprosessin vaiheessa todennetaan, että tämä malli esiintyy laajemmissa samantyyppisiä tietoja sisältävässä aineistossa. Kaikki nämä kolme vaihetta on kuvattu tässä väitöskirjassa. Lisäksi on esitetty useita tiedonlouhinta-algoritmeja ja menetelmiä.

Toisessa osassa esitetään useita matkaviestinverkkoihin ja verkon turvallisuuteen liittyviä tapaustutkimuksia. Jokaisessa esimerkkitapauksessa sovelletaan yhtä tai useampaa tiedonlouhintatekniikkaa, jotta voidaan ratkaista esitettyjä ongelmia. Ensin käytetään geneettistä algoritmia toistimien optimaalisten koordinaattien löytämiseksi WiMAX-verkoissa. Seuraavaksi liikkuvan käyttäjän seuraavan paikan ennustaminen suoritetaan soveltamalla useita luokittelumenetelmiä käyttäjän spatiotemporaalisiin liikeratoihin. Sitten sovelletaan useita klusterointi- ja poikkeamienhavaitsemisalgoritmeja, jotta havaitaan epätavallisia HTTP-pyyntöjä. Lopuksi, tiedonlouhintamenetelmää käytetään haittaohjelmien havaitsemiseen ja luokitteluun.

REFERENCES

[AAN11] R.M. Alguliev, R.M. Aliguliyev, and S.A. Nazirova. Classification of textual e-mail spam using data mining techniques. *Appl. Comp. Intell. Soft Comput.*, pages 10:10–10:10, 2011.

[ABR64] M.A. Aizerman, E.A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control*, number 25 in Automation and Remote Control, pages 821–837, 1964.

[AGA10] H. Abu-Ghazaleh and A.S. Alfa. Application of mobility prediction in wireless networks using markov renewal theory. *IEEE Tran. on Vehicular Technology*, 59:788–802, 2010.

[Agr03] D.K. Agrafiotis. Stochastic proximity embedding. *Journal of Computational Chemistry*, 24(10):1215–1221, 2003.

[AI02] S. Abe and T. Inoue. Fuzzy support vector machines for multiclass problems. In *Proceedings of the European Symposium on Artificial Neural Networks*, pages 113–118, 2002.

[AIJ11] G.A. Abed, M. Ismail, and K. Jumari. Traffic modeling of lte mobile broadband network based on ns-2 simulator. In *Proceedings of the 3rd International Conference on Computational Intelligence, Communication Systems and Networks*, CICSYN '11, pages 120–125, 2011.

[Alb03] S. Albayrak. Unsupervised clustering methods for medical data: An application to thyroid gland data. In *Proceedings of the Joint International Conference on Artificial Neural Networks and Neural Information Processing*, pages 695–701, 2003.

[Ber02] P. Berkhin. Survey of clustering data mining techniques. Technical report, 2002.

[BFOS84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and regression trees*. Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, USA, 1984.

[BG05] I. Borg and P. J. F. Groenen. *Modern Multidimensional Scaling*. Springer-Verlag, New York, USA, 2005.

[BKNS00] M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *Proceedings of the 2000 ACM Sigmoid International Conference on Management of Data*, pages 93–104. ACM, 2000.

[BL02] H. Byun and S.-W. Lee. Applications of support vector machines for pattern recognition: A survey. In *Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines*, SVM '02, pages 213–236, 2002.

[BMvL12] S. Bubeck, M. Meila, and U. von Luxburg. How the initialization affects the stability of the k-means algorithm. *ESAIM: Probability and Statistics*, 16:436–452, 2012.

[BN03] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[BP10] R. Battiti and A. Passerini. Brain-computer evolutionary multiobjective optimization (bc-emo): a genetic algorithm adapting to the decision maker. *IEEE Transactions on Evolutionary Computation*, 14(15):671–687, 2010.

[CBK09] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[CCLC09] C.-Y. Chang, C.-T. Chang, M.-H. Li, and C.-H. Chang. A novel relay placement mechanism for capacity enhancement in ieee 802.16j wimax networks. In *Proceedings of the IEEE International Conference on Communications*, pages 5201–5205, 2009.

[CD07] P. Cunningham and S.J. Delany. k-nearest neighbour classifiers, 2007.

[CEF$^+$06] S. Chakrabarti, M. Ester, U. Fayyad, J. Gehrke, J. Han, S. Morishita, G. Piatetsky-Shapiro, and W. Wang. Data mining curriculum. A proposal by acm sigkdd, 2006.

[CG10] I. Corona and G. Giacinto. Detection of server-side web attacks. In *Proceedings of the JMLR: Workshop on Applications of Pattern Analysis*, pages 160–166, 2010.

[CHS$^+$01] D.J. Cook, L.B. Holder, S. Su, R. Maglothin, and I. Jonyer. Structural mining of molecular biology data. *IEEE Transactions on Evolutionary Computation*, 20(4):67–74, 2001.

[CLL$^+$05] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, F. Warner, and S. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. In *Proceedings of the National Academy of Sciences*, pages 7426–7431, 2005.

[CML11] E. Cho, S.A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1082–1090, 2011.

[CP02] A. Chan and E. Pampalk. Growing hierarchical self organising map (ghsom) toolbox: visualisations and enhancements. In *Proceedings of the 9-th International Conference Neural Information Processing*, volume 5, pages 2537–2541, Nov 2002.

[CSS$^+$07] B. Cao, D. Shen, J.-T. Sun, Q. Yang, and Z. Chen. Feature selection in a kernel space. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 121–128, New York, NY, USA, 2007. ACM.

[CSZ10] O. Chapelle, B. Schlkopf, and A. Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

[CY06] H. Chang and D.-Y. Yeung. Robust locally linear embedding. *Pattern Recognition*, 39(6):1053–1065, 2006.

[DHCW10] J. Deng, J. Hu, H. Chi, and J. Wu. An improved fuzzy clustering method for text mining. In *Proceedings of the 2nd International Conference on Networks Security, Wireless Communications and Trusted Computing*, volume 1, pages 65–69, 2010.

[DMR00] M. Dittenbach, D. Merkl, and A. Rauber. The growing hierarchical self-organizing map. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, volume 6, pages 15–19, 2000.

[DYY13] V. Dewan, N. Yadav, and N. Yadav. Knowledge discovery in databases (data mining): a perspective. *Discovery Engineering*, 2(8):69–73, 2013.

[EKJX96] M. Ester, H. Kriegel, S. Jörg, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[ET94] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Taylor & Francis, United Kingdom, 1994.

[FGG97] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2–3):131–163, 1997.

[Fis38] R.A. Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics*, 8(4):376–386, 1938.

[For65] E.W. Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[FPSS96] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, 39(11):27–34, November 1996.

[Fri96] J. Friedman. Another approach to polychotomous classification. Dept. statistics, stanford univ., 1996.

[FSHS10] Y. Fukushima, A. Sakai, Y. Hori, and K. Sakurai. A behavior based malware detection scheme for avoiding false positive. In *Proceedings of the 6th IEEE Workshop on Secure Network Protocols (NPSec)*, pages 79–84, 2010.

[GLJ$^+$11] Yu Gong, Yong Li, Depeng Jin, Li Su, and Lieguang Zeng. A location prediction scheme based on social correlation. In *Proceedings of the IEEE 73rd Vehicular Technology Conference (VTC Spring)*, pages 1–5, 2011.

[Gol06] D. Gollmann. *Computer Security*. Wiley, 2nd edition, 2006.

[GRHS04] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2004.

[GRS$^+$09] X. Guo, R. Rouil, C. Soin, S. Parekh, B. Sikdar, and S. Kalyanaraman. Wimax system design and evaluation methodology using the ns-2 simulator. In *Proceedings of the 1st International Conference on COMmunication Systems And NETworks*, COMSNETS'09, pages 643–652, 2009.

[GSCJ13] M. Gupta, A. B. Sharma, H. Chen, and G. Jiang. Context-aware time series anomaly detection for complex systems. In *Proceedings of the SDM Workshop on Data Mining for Service and Maintenance*, 2013.

[GT12] F. Gullo and A. Tagarelli. Uncertain centroid based partitional clustering of uncertain data. *Proc. VLDB Endow.*, 5(7):610–621, 2012.

[GTDVMFV09] P. García-Teodoroa, J. Díaz-Verdejoa, G. Maciá-Fernándeza, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1–2):18–28, 2009.

[GTL12] H. Gao, J. Tang, and H. Liu. Mobile location prediction in spatio-temporal context. In *Procedings of Mobile Data Challenge by Nokia Workshop at the Tenth International Conference on Pervasive Computing*, pages 1–4, 2012.

[GX09] Wang Guangming and Wang Xiaoliang. Application of classification algorithm based on association rules in the labor market. In *Proceedings of the International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government (EEEE)*, pages 255–258, 2009.

[Hay98] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2 edition, New Jersey, USA, 1998.

[HBN11] N. Hubballi, S. Biswas, and S. Nandi. Sequencegram: n-gram modeling of system calls for program based anomaly detection. In *Proceedings of the 3rd International Conference on Communication Systems and Networks (COMSNETS)*, pages 1–10, 2011.

[HCY10] Guang-Xin Huang, Hua-Fu Chen, and Feng Yin. Improved support vector data description. In *Proceedings of the International Conference on Machine Learning and Cybernetics (ICMLC)*, volume 3, pages 1459–1463, 2010.

[Her99] J.W. Herrmann. A genetic algorithm for minimax optimization problems. In *Proceedings of the Congress on Evolutionary Computation*, pages 1099–1103, 1999.

[HGN00] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining &mdash; a general survey and comparison. *SIGKDD Explor. Newsl.*, 2(1):58–64, June 2000.

[HH89] E.L. Hines and R.A. Hutchinson. Application of multi-layer perceptrons to facial feature location. In *Proceedings of the 3rd International Conference on Image Processing and its Applications*, pages 39–43, 1989.

[HH09] C.F. Hsu and H.F. Hung. Classification methods of credit rating - a comparative analysis on svm, mda and rst. In *Proceedings of the International Conference on Computational Intelligence and Software Engineering (CiSE)*, pages 1–4, 2009.

[HPK09] T. Hirsimaki, J. Pylkkonen, and M. Kurimo. Importance of high-order n-gram models in morph-based speech recognition. *Audio, Speech, and Language Processing, IEEE Transactions*, 17:724–732, May 2009.

[HR] G. Hinton and S. Roweis. Stochastic neighbor embedding. In *Advances in Neural Information Processing Systems 15*, pages 833–840. MIT Press.

[HS85] D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

[HYLZ09] T. Huang, Y. Yu, K. Li, and W. Zeng. Reckon the parameter of dbscan for multi-density data sets with constraints. In *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence*, volume 4, pages 375–379, 2009.

[Jai10] A.K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett.*, 31(8):651–666, 2010.

[Jol86] I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, USA, 1986.

[JS12] A. Juvonen and T. Sipola. Adaptive framework for network traffic classification using dimensionality reduction and clustering. In *Proceedings of the 4th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pages 274–279, 2012.

[JYTK11] F. Jiang, J. Yuan, A.S. Tsaftaris, and A.K. Katsaggelos. Anomalous video event detection using spatiotemporal context. *Comput. Vis. Image Underst.*, 115(3):323–333, 2011.

[Kim11] J. Kim. The anomaly detection by using dbscan clustering with multiple parameters. In *Proceedings of the ICISA*, pages 1–5, 2011.

[Kis04] Özgür Kisi. Multi-layer perceptrons with levenberg-marquardt training algorithm for suspended sediment concentration prediction and estimation / prévision et estimation de la concentration en matières en suspension avec des perceptrons multi-couches et l'algorithme d'apprentissage de levenberg-marquardt. *Hydrological Sciences Journal*, 49(6), 2004.

[KKP06] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas. Data preprocessing for supervised leaning. *International Journal of Computer Science*, 1(2):111–117, 2006.

[KLS09] Gun-Woo Kim, Jeong Hwa Lee, and Jin Hyun Son. Classification and analyses of business process anomalies. In *Proceedings of the International Conference on Communication Software and Networks (ICCSN)*, pages 433–437, 2009.

[Kor07] M.F. Korns. In R. Riolo, T. Soule, and B. Worzel, editors, *Genetic Programming Theory and Practice V*, Genetic and Evolutionary Computation, chapter 4, pages 53–68. Springer, 2007.

[Kot07] S.B. Kotsiantis. Supervised machine learning: A review of classification techniques. In *Proceedings of the Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pages 3–24, 2007.

[KP04] S. B. Kotsiantis and P. E. Pintelas. Recent advances in clustering: A brief survey. *WSEAS Trans. Inform. Sci. Appl*, pages 73–81, 2004.

[KP09] S. Kotsiantis and P. Pintelas. *Predictive Data Mining: A Survey of Regression Methods*. IGI Global, Hershey, USA, 2009.

[KP12] Y. Khaokaew and S. Pukkawanna. Time series anomaly detection using recessive subsequence. In *Proceedings of the The International Conference on Information Network*, ICOIN '12, pages 329–333, 2012.

[KR92] K. Kira and L. Rendell. The feature selection problem: traditional methods and new algorithm. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 129–134, 1992.

[KS13] G. Kesavaraj and S. Sukumaran. A study on classification techniques in data mining. In *Proceedings of the 4th International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–7, 2013.

[KSRS97] Igor Kononenko, Edvard Simec, and Marko Robnik-Sikonja. Overcoming the myopia of inductive learning algorithms with relieff. *Applied Intelligence*, 7:39–55, 1997.

[KV03] C. Kruegel and G. Vigna. Anomaly detection of web-based attacks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, CCS '03, pages 251–261, 2003.

[LDb10] L. Lin and X. De-bao. Research on the network security management based on data mining. In *Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, pages 184–187, 2010.

[LK12] Li Long and K. Kianmehr. Internet traffic classification based on associative classifiers. In *Proceedings of the IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 263–268, 2012.

[LKFH05] J. Lin, E. Keogh, A. Fu, and H.V. Herle. Approximations to magic: Finding unusual medical time series. In *Proceedings of the 18th IEEE Symp. on Computer-Based Medical Systems (CBMS)*, pages 23–24, 2005.

[Llo06] S. Lloyd. Least squares quantization in pcm. *IEEE Trans. Inf. Theor.*, 28(2):129–137, 2006.

[Loh12] S. Lohr. The age of big data. *New York Times 11*, 2012.

[LP14] W.-K. Loh and Y.-H. Park. A survey on density-based clustering algorithms. *Lecture Notes in Electrical Engineering*, 280:775–780, 2014.

[LR05] P. Lehtimäki and K. Raivio. A som based approach for visualization of gsm network performance data. In *Proceedings of the 18th International Conference on Innovations in Applied Artificial Intelligence*, IEA/AIE'2005, pages 588–598, 2005.

[LSPS11] P.-R. Lei, T.-J. Shen, W.-C. Peng, and I.-J. Su. Exploring spatial-temporal trajectory model for location prediction. In *Proceedings of the IEEE 12th International Conference on Mobile Data Management*, volume 1 of *MDM '11*, pages 58–67, 2011.

[LWSH05] W. Li, K. Wang, S. Stolfo, and B. Herzog. Fileprints: Identifying file types by n-gram analysis. In *Proceedings from the 6th Annual IEEE Systems, Man, and Cybernetics (SMC) Information Assurance Workshop*, pages 64–71, 2005.

[LZO04] T. Li, C. Zhang, and M. Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20(15):2429–2437, October 2004.

[MBM$^+$12] A. Miller, G. Barwell, G. McClymont, S. McPartland, S. Metcalfe, D. Morris, S. Mosley, P. Nash, J. Reynolds, G. Stringer, and R. Williams. Malware and cyber crime. House of commons - science and technology committee, 12th report of session 2010-12, 2012.

[MBN02] L.C. Molina, L. Belanche, and A. Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of the IEEE International Conference on Data Mining*, ICDM '02, pages 306–313, Washington, DC, USA, 2002. IEEE Computer Society.

[MCW04] L. Ming, Y. Cheung, and Y. Wang. A dynamically switched crossover for genetic algorithms. In *Proceedings of the International Conference on Machine Learning and Cybernetics*, pages 3254–3257, 2004.

[MD09] Li Min and Wang Dongliang. Anormaly intrusion detection based on som. In *Proceedings of the WASE International Conference on Information Engineering*, volume 1, pages 40–43, 2009.

[MGSK88] H. Muhlenbein, M. Georges-Schleuter, and O. Kramer. Evolution algorithms in combinatorial optimization. *Parallel Computing*, 7(1):65–85, 1988.

[MMR$^+$01] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[MRW$^+$99] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels, 1999.

[NHKH09] D. Niyato, E. Hossain, D.I. Kim, and Z. Han. Joint optimization of placement and bandwidth reservation for relays in ieee 802.16j mobile multihop networks. In *Proceedings of the IEEE International Conference on Communications*, ICC'09, pages 4843–4847, 2009.

[NLLS11] S. Novakov, C.-H. Lung, I. Lambadaris, and N. Seddigh. Studies in applying pca and wavelet algorithms for network traffic anomaly detection. In *Proceedings of the Workshop on Building Anal. Datasets & Gathering Exp. Returns for Security*, pages 29–36, 2011.

[NT10] M. H. Nguyen and F. Torre. Optimal feature selection for support vector machines. *Pattern Recogn.*, 43(3):584–591, 2010.

[NTGG$^+$05] A. Nguyen-Tuong, S. Guarnieri, D. Greene, J. Shirley, and D. Evans. Automatically hardening web applications using precise tainting. In *Procedings of the 20th IFIP International Information Security Conference*, 2005.

[PG93] W. Punch and E. Goodman. Further research on feature selection and classification using genetic algorithms. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 557–564, 1993.

[PG13] K.A. Piirainen and R.A. Gonzalez. Seeking constructive synergy: Design science and the constructive research approach. In *Proceedings of the 8th International Conference on Design Science at the Intersection of Physical and Virtual Design*, DESRIST'13, pages 59–72, 2013.

[PH09] Steven W. Peters and Robert W. Heath. The future of wimax: Multihop relaying with ieee 802.16j. *Comm. Mag.*, 47(1):104–111, 2009.

[PJ13] H. Pereira and E. Jamhour. A clustering-based method for intrusion detection in web servers. In *Proceedings of 20th International Conference on Telecommunications (ICT)*, pages 1–5, 2013.

[PSF91] G. Piatetsky-Shapiro and W. J. Frawley. *Knowledge Discovery in Databases*. AAAI/MIT Press, Cambridge, USA, 1991.

[PSN05] D.T. Pham, S.S. Dimov, and C.D. Nguyen. Selection of k in k-means clustering. In *Proceedings of IMechE Mechanical Engineering Science*, pages 103–119, 2005.

[PT10] B. Pardeshi and D. Toshniwal. Improved k-medoids clustering based on cluster validity index and object density. In *Proceedings of the 2nd IEEE International Advance Computing Conference (IACC)*, pages 379–384, 2010.

[PTC13] C. Ping-Tsai and S.H. Chung. On data integration and data mining for developing business intelligence. In *Proceedings of Conference on Systems, Applications and Technology Conference (LISAT)*, pages 1–6, 2013.

[QMG12] K. Qazanfari, M.S. Mirpouryan, and H. Gharaee. A novel hybrid anomaly based intrusion detection method. In *Proceedings of the 6th International Symposium on Telecommunications (IST)*, pages 942–947, 2012.

[RHB11] D. Rupprecht, S. Hesse, and R. Blum. Automatic face feature points extraction. In *Proceedings of the 3rd International Conference on Digital Human Modeling*, ICDHM'11, pages 186–194. Springer-Verlag, 2011.

[Rie94] M. Riedmiller. Advanced supervised learning in multi-layer perceptrons — from backpropagation to adaptive learning algorithms. *Computer Standards & Interfaces*, 16(3):265–278, 1994.

[RMD02] A. Rauber, D. Merkl, and M. Dittenbach. The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, 13:1331–1341, 2002.

[Rom97] G. Romp. *Game Theory: Introduction and Applications*. Oxford University Press, Oxford, United Kingdom, 1997.

[ROT03] M. Ramadas, S. Ostermann, and B. Tjaden. Detecting anomalous network traffic with self-organizing maps. *Recent Advances in Intrusion Detection*, pages 36–54, 2003.

[RP05] N. Retnakaran and N.J. Pizzi. Biomedical pattern classification using an optimized fuzzy adaptive logic network. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, pages 382–385, 2005.

[RS02] N. Robinson and M. Shapcott. Data mining information visualisation - beyond charts and graphs. In *Proceedings of the 6th International Conference on Information Visualisation*, pages 577–583, 2002.

[RS11] C. Ramaraju and N. Savarimuthu. A classification model for customer segmentation. *Advances in Computing and Communications: Communications in Computer and Information Science*, 190:661–670, 2011.

[RTR$^+$01] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. Mesirov, T. Poggio, W. Gerald, M. Loda, E. Lander, and T. Golub. Multiclass cancer diagnosis using tumor gene expression signatures. *Proceedings of National Academy of Sciences of the USA*, 98(26):15149–15154, 2001.

[RVC12] M.K. Rafsanjani, Z.A. Varzaneh, and N.E. Chukanlo. A survey of hierarchical clustering algorithms. *The Journal of Mathematics and Computer Science*, 5(3):229–240, 2012.

[SAM$^+$11] Alexander Sayenko, Olli Alanen, Henrik Martikainen, Vitaliy Tykhomyrov, Oleksandr Puchko, Vesa Hytönen, and Timo Hämäläinen. Winse: Wimax ns-2 extension. *Simulation*, 87(1-2):24–44, 2011.

[SBUPB13] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P.G. Bringas. Opcode sequences as representation of executables for data-mining-based unknown malware detection. *Inf. Sci.*, 231:64–82, 2013.

[SH95] R. Setiono and L. C. K. Hui. Use of a quasi-newton method in a feedforward neural network construction algorithm. *IEEE Transactions on Neural Networks*, 6(1):273–277, 1995.

[Sib73] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[Ski07] D.B. Skillicorn. Detecting anomalies in graphs. In *Proceedings of the IEEE Conference on Intelligence and Security Informatics*, pages 209–216, 2007.

[SKLM05] T. Shon, Y. Kim, C. Lee, and J. Moon. A machine learning framework for network anomaly detection using svm and ga. In *Proceedings of the 6th Annual IEEE SMC*, pages 176–183, 2005.

[SL06] R. Sharapov and A. Lapshin. Optimal feature selection for support vector machines. *Pattern Recognition and Image Analysis*, 16(3):392–397, 2006.

[SLZ01] S. Shekhar, C.-T. Lu, and P. Zhang. Detecting graph-based spatial outliers: Algorithms and applications (a summary of results). In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 371–376, 2001.

[SM09] C. Saranya and G. Manikandan. A study on normalization techniques forprivacy preserving data mining. *International Journal of Engineering and Technology (IJET)*, 5(3):2701–2704, May 2009.

[SM11] Michael R. Smith and Tony Martinez. Improving classification accuracy by identifying and removing instances that should be misclassified. In *Proceedings of the International Joint Conference on Neural Networks*, pages 2690–2697, 2011.

[Smi12] A. Smiti. Dbscan-gm: An improved clustering method based on gaussian means and dbscan techniques. In *Proceedings of the IEEE 16th International Conference on Intelligent Engineering Systems (INES)*, pages 573–578, 2012.

[SMJ02] J. Souza, S. Matwin, and N. Japkowicz. Evaluating data mining models: A pattern language, 2002.

[SSB11] N.V. Sawant, K. Shah, and V.A. Bharadi. Survey on data mining classification techniques. In *Proceedings of the International Conference &#38; Workshop on Emerging Trends in Technology*, ICWET '11, pages 1380–1380, New York, NY, USA, 2011. ACM.

[Sue79] C. Y. Suen. n-gram statistics for natural language understanding and text processing. *Pattern Analysis and Machine Intelligence, IEEE Transactions*, PAMI-1:162–172, Nov 1979.

[SWL08] M. Siddiqui, M.C. Wang, and J. Lee. A survey of data mining techniques for malware detection using file features. In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ACM-SE 46, pages 509–510, 2008.

[SWW08] R. Schoenen, W.Zirwas, and B.H. Walke. Raising coverage and capacity using fixed relays in a realistic scenario. In *Proceedings of the 14th European Wireless Conference*, pages 1–6, 2008.

[SZWS09] C. Shao, X. Zhang, C. Wan, and W. Shang. A som-based method for manifold learning and visualization. In *Proceedings of the International Joint Conference on Computational Sciences and Optimization*, volume 2, pages 312–316, 2009.

[TD04] D.M.J. Tax and R.P.W. Duin. Support vector data description. *Machine Learning*, 54(1):45–66, 2004.

[TSD09] M. Tuma, S. Scholz, and R. Decker. The application of cluster analysis in marketing research: A literature analysis. *Business Quest Journal*, 14, 2009.

[TTLC09] X. Tao, S. Thummalapenta, D. Lo, and L. Chao. Data mining for software engineering. *Computer*, 42(8):55–62, 2009.

[TV05] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, New York, NY, USA, 2005.

[Ult05] A. Ultschk. Clustering with som: U*c. In *Proceedings of the Workshop on Self-Organizing Maps (WSOM)*, pages 75–82, 2005.

[WRG$^+$10] N. Wisittipanit, H. Rangwala, P. Gillevet, M. Sikaroodi, E. A. Mutlu, and A. Keshavarzian. Svm-based classification and feature selection methods for the analysis of inflammatory bowel disease microbiome data. In *Proceedings of the 9th International Workshop on Data Mining in Bioinformatics*, pages 1–8, 2010.

[WS04] K.Q. Weinberger and L.K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 988–995, 2004.

[WS09] K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, 2009.

[WSH$^+$10] W. Wang, H. Sharif, M. Hempel, T. Zhou, B. Wysocki, and T. Wysocki. Implementation and performance evaluation of qos scheduling algorithms in mobile wimax ns-2 simulator. In *Proceedings of the 4th International Conference on Signal Processing and Communication Systems (ICSPCS)*, pages 1–6, 2010.

[WX11] Xin Wang and Yaxi Xu. Detecting anomalies in symbolic sequence dataset. In *Proceedings of the International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)*, pages 443–447, 2011.

[XKB$^+$05] X. Xiong, Y. Kim, Y. Baek, D.W. Rhee, and S.-H. Kim. Analysis of breast cancer using data mining and statistical techniques. In *Proceedings of the 6th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks*, pages 82–87, 2005.

[XYYP08] C. Xiaoyun, M. Yufang, Z. Yan, and W. Ping. Gmdbscan: Multi-density dbscan cluster based on grid. In *Proceedings of the IEEE International Conference on e-Business Engineering*, pages 780–783, 2008.

[Yan02] M.-H. Yang. Extended isomap for pattern classification. In *Eighteenth National Conference on Artificial Intelligence*, pages 224–229, 2002.

[YD11] C. Yang and T. Dorbin. Analyzing and visualizing web opinion development and social interactions with density-based clustering. *Trans. Sys. Man Cyber. Part A*, 41(6):1144–1155, 2011.

[YHGL10] Yun Yang, Guyu Hu, Shize Guo, and Jun Luo. Imbalanced classification algorithm in botnet detection. In *Proceedings of the 1st International Conference on Pervasive Computing Signal Processing and Applications (PCSPA)*, pages 116–119, 2010.

[YS02] Liew Pol Yee and L.C. De Silva. Application of multilayer perceptron networks in public key cryptography. In *Proceedings of the the International Joint Conference on Neural Networks*, pages 1439–1443, 2002.

[ZC09] Q. Zhang and S. Chang. An improved crossover operator of genetic algorithm. In *Proceedings of the 2nd International Symposium on computational Intelligence and Design*, pages 82–86, 2009.

[ZS09] M. Zhao and V. Saligrama. Anomaly detection with score functions based on nearest neighbor graphs. In *Proceedings of Advances in Neural Information Processing Systems*, volume 22, pages 2250–2258, 2009.

ORIGINAL PAPERS

PI

OPTIMAL RELAYS DEPLOYMENT FOR 802.16J NETWORKS

by

Mikhail Zolotukhin, Vesa Hytönen, Timo Hämäläinen and Andrey Garnaev 2012

Springer. Mobile Networks and Management. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 97, pp. 31–45

PII

A RELAY DEPLOYMENT MECHANISM FOR ONE SCENARIO OF COST-EFFECTIVE COVERAGE EXTENSION IN IEEE 802.16J NETWORKS

by

Mikhail Zolotukhin, Timo Hämäläinen and Andrey Garnaev 2012

Proceedings of the 5th International Conference on New Technologies, Mobility
and Security (NTMS), pp. 1–6

PIII

ONLINE ANOMALY DETECTION BY USING N-GRAM MODEL AND GROWING HIERARCHICAL SELF-ORGANIZING MAPS

by

Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen 2012

Proceedings of the 8th International Wireless Communications and Mobile
Computing Conference (IWCMC), pp. 47–52

PIV

GROWING HIERARCHICAL SELF-ORGANIZING MAPS AND STATISTICAL DISTRIBUTION MODELS FOR ONLINE DETECTION OF WEB ATTACKS

by

Mikhail Zolotukhin, Timo Hämäläinen and Antti Juvonen 2013

Springer. Web Information Systems and Technologies. Lecture Notes in Business
Information Processing, Vol. 140, pp. 281–295

PV

NOVEL METHOD FOR THE PREDICTION OF MOBILE LOCATION BASED ON TEMPORAL-SPATIAL BEHAVIORAL PATTERNS

by

Mikhail Zolotukhin, Elena Ivannikova and Timo Hämäläinen 2013

Proceedings of the 3rd IEEE International Conference on Information Science
and Technology (ICIST), pp. 761–766

PVI

DETECTION OF ANOMALOUS HTTP REQUESTS BASED ON ADVANCED N-GRAM MODEL AND CLUSTERING TECHNIQUES

by

Mikhail Zolotukhin and Timo Hämäläinen 2013

Springer. Internet of Things, Smart Spaces, and Next Generation Networking.
Lecture Notes in Computer Science, Vol. 8121, pp. 371–382

PVII

SUPPORT VECTOR MACHINE INTEGRATED WITH GAME-THEORETIC APPROACH AND GENETIC ALGORITHM FOR THE DETECTION AND CLASSIFICATION OF MALWARE

by

Mikhail Zolotukhin and Timo Hämäläinen 2013

Proceedings of Globecom 2013 Workshop - the 1st International Workshop on
Security and Privacy in Big Data, pp. 211–216

PVIII

DETECTION OF ZERO-DAY MALWARE BASED ON THE ANALYSIS OF OPCODE SEQUENCES

by

Mikhail Zolotukhin and Timo Hämäläinen 2014

Proceedings of the 11th Annual IEEE Consumer Communications and
Networking Conference (CCNC), pp. 677–682