Feature Selection for Intrusion Detection System

Jingping Song

Supervisors: Prof. Chris Price
Prof. Qiang Shen

Ph.D. Thesis

Department of Computer Science
Institute of Mathematics, Physics and Computer Science
Aberystwyth University

March 3, 2016

Declaration and Statement

DECLARATION

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ..... (candidate)

Date .....

STATEMENT 1

This thesis is the result of my own investigations, except where otherwise stated. Where **correction services**[1] have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ..... (candidate)

Date .....

STATEMENT 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..... (candidate)

Date .....

[1] This refers to the extent to which the text has been corrected by others.

Abstract

Intrusion detection is an important task for network operators in today's Internet. Traditional network intrusion detection systems rely on either specialized signatures of previously seen attacks, or on labeled traffic datasets that are expensive and difficult to reproduce for user-profiling to hunt out network attacks. Machine learning methods could be used in this area since they could get knowledge from signatures or as normal-operation profiles. However, there is usually a large volume of data in intrusion detection systems, for both features and instances.

Feature selection can be used to optimize the classifiers used to identify attacks by removing redundant or irrelevant features while improving the quality. In this thesis, six feature selection algorithms are developed, and their application to intrusion detection is evaluated.

They are: Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm, New Evidence Accumulation Ensemble with Hierarchical Clustering Algorithm, Modified Mutual Information-based Feature Selection Algorithm, Mutual Information-based Feature Grouping Algorithm, Feature Grouping by Agglomerative Hierarchical Clustering Algorithm, and Online Streaming Feature Selection Algorithm.

All algorithms are evaluated on the KDD 99 dataset, the most widely used data set for the evaluation of anomaly detection methods, and are compared with other algorithms. The potential application of these algorithms beyond intrusion detection is also examined and discussed.

Acknowledgements

I would like to express my thanks for the great support from many people, without whom this thesis would not have been possible.

Firstly, I would like to express uttermost gratitude to my supervisor Prof. Chris Price for his guidance, advice and effort, which have been essential at all stages of my research and shaped the thesis.

Also I am extremely grateful to my second supervisor Prof. Qiang Shen for his insightful advice and constant inspiration in this research. I would like to express my deepest appreciation to Dr Richard Jensen and Dr. Neil S. Mac Parthaláin for the stimulating discussions and helpful advice. Besides, my sincere gratitude goes to my examiners for their time and effort, which will improve this thesis in various ways.

I would like to thank my fellow researchers in the Advanced Reasoning Group for the discussions, inspiration and team work. They are Dr Ren Diao, Dr Nitin Naik, Dr Chengyuan Chen, Dr Shangzhu Jin, Dr Pan Su, Liang Shen, Yongfeng Zhang, Tianhua Chen, Ling Zheng and Zhenpeng Li. My thanks also go to Computer Science Department for their assistance and comfort that makes me feel like being home.

Also I am very thankful to my friends Diana Whitehouse, David Whitehouse, Dr Ran Song, Dr Zhili Chen, Dr Ziming Zeng, Dr Yitian Zhao, Dr Chen Gui, Dr Lilan Pan, Dr Jingrong Long, Dr Min Zhang, Dr Minfeng Huang, Lu Lou, Liping Wang, Juan Cao, Peter Scully for the friendship, inspiration and encouragement during my research.

Finally and most importantly, I would like to thank my mother Wenhui Chen and my wife Ni Zhu for their constant support, encouragement and motivation, which enabled me to overcome any difficulties I encountered during my research.

At last thanks go to my family members Jubilee and Millie, and their friends Miu Miu and Jasper for their companion and the great happiness they brought.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

With the development of computer technology and network communication technology, computer network spread rapidly in recent years. Internet has become an important medium for information exchange and sharing in our society. Internet has huge information capacity, high speed transmission, worldwide coverage, a high degree of openness and interactivity. So it is profoundly changing the way of people's work and live, and influences political, economic, military, cultural and technological development[126].

Network information security has become more and more serious as the rapid development of the computer network. And it is also an important factor restricting the development of the network. In recent years, network attacks and information security incidents occured frequently, covering areas more and more widely, and increasingly harmful[164]. In october of 2002, the top 13 root domain name server which are responsible for global Internet working are attacked by DDoS(Distributed Denial of Service). And it results in nine servers were interrupted their service[157]. In January 2003, Internet suffered massive "worm" virus. Global network services were severely affected, including the Americas, Europe, Asia, Australia[26]. From 2011, other new challenges appeared, such as data unauthorized disclosure, cell phone privacy and security problems, Advanced Persistent Threats (APT) attacks and so on. Network security issues have caused economic losses to some companies. And users will panic and dare not trust the network[130, 5]. Network security is a process of defence all forms of threats from internal and external in order to ensure security of communications network and information [60]. And network security

design usually include safety equipment, firewall, vitual private network, intrusion detection system, security server and access control mechanism. According to system security, research on network security is divided into attack and defense. Attack techniques include computer network scanning, monitoring, stealth, invasion and backdoor. Defense technology mainly includes security configuration of operating system, firewall technology, information encryption technology and network intrusion detection[156].

1.1 Network Security Threat

Network security covers secure storage, transport and application of information in network[16]. The concept of network security mainly includes the following basic elements which is shown in figure 1.1.

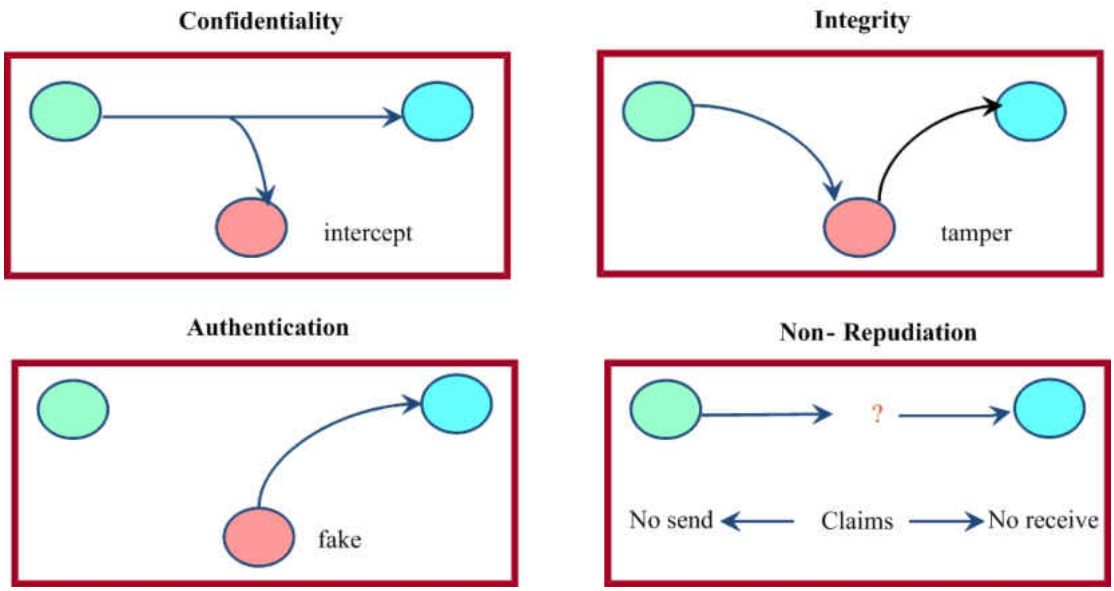

Figure 1.1: Basic elements of network security

Confidentiality means that information is not leaked to unauthorized entities. And unauthorized entities could not access and understand the information. Integrity is concerned with prevent information from unauthorized tampering or destruction in order to ensure that the system the information provided is complete and correct. Authentication verifies the authenticity of the identity declared at the identification stage. Non-repudiation refers to prove that the message was sent or receive by

using cryptographic digital signatures[171]. Besides the four elements in figure 1.1, availability, controllability and accountability are also elements of network security.

At present, there are many threats in network and they are mainly in following aspects.

1. Unauthorized access. Access to the network or computer information resources without pre-authorized.
2. Destroy the integrity of information. Delete, modify or add some important information to interfere with the normal use of the information in the system, or to obtain a response which is attackers desired .
3. Information leakage or loss. Information is lost or stolen in transit, information loss or leakage in a storage medium, the encrypted information is decrypted and so on.
4. Interfere with the normal operation of the system. Constantly on the network service system interference, to change its normal operating state, make the system response slow down or even stop, affecting the normal users.

Currently, computer viruses and network intrusion attacks are the two most common implementation of network threats. Network attacks are always in a large number of network activities and not susceptible to geographical and time limits.

Network attacks can be classified from different ways[147]. According to the location and manner of attacks, network attacks can be divided into remote attacks, local attacks and pseudo remote attacks. In accordance with the purpose and general characteristics of attacks, the current network attacks can be classified to denial of service attack, information collection attacks and so on. The computer system attack classification based on the known attack classification analysis and personal knowledge was composed by N. Paulauskas and E. Garsva in 2006[146]. The suggested computer system attack classification is shown in figure 1.2. Every attack possesses all 14 listed features.

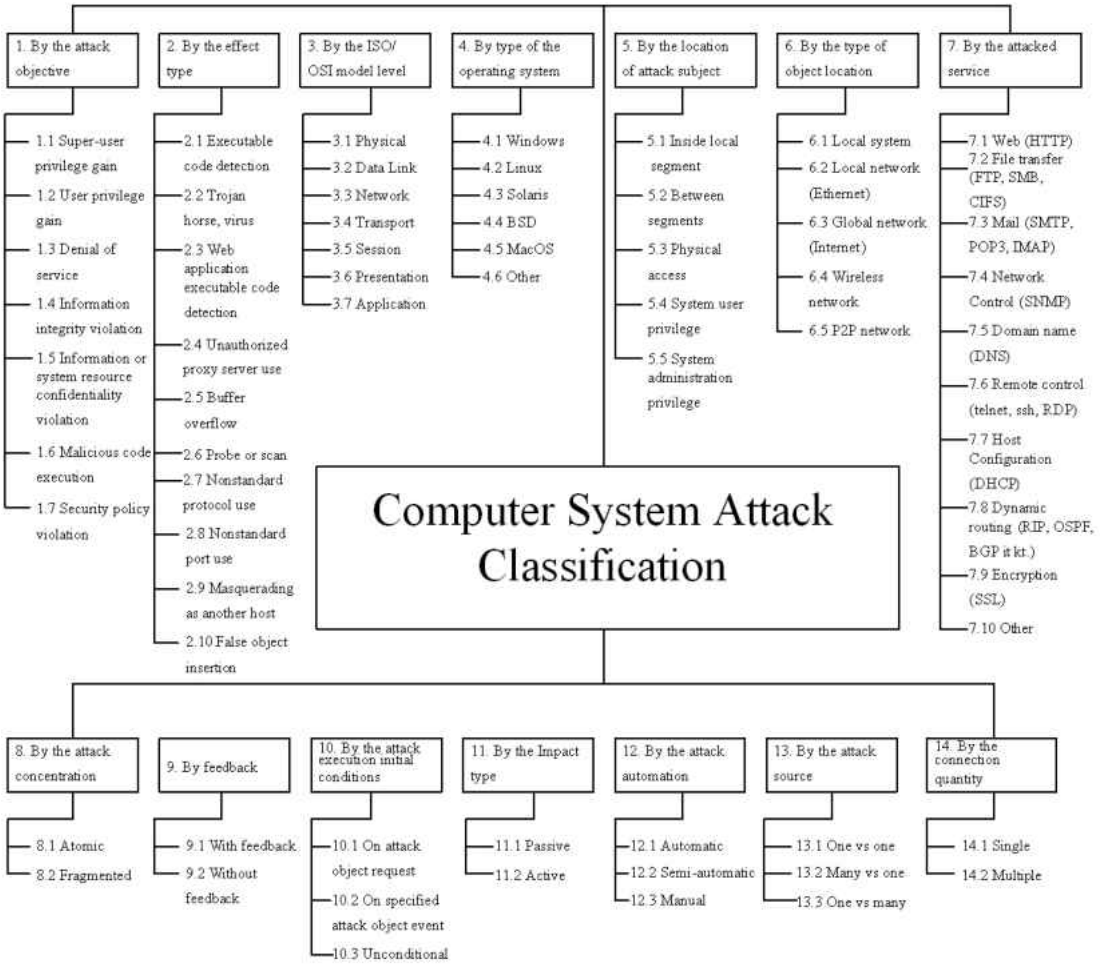

Figure 1.2: Suggested computer system attack classification

1.2 Intrusion Detection System and Feature Selection

1.2.1 Intrusion Detection System

As it is described above, intrusion detection system (IDS) is a part of network security design. An intrusion detection system monitors network traffic for suspicious activity and alerts the system or network administrator in order to take evasive action. It has a very important position in the network information security and it is considered as the second security gate after firewall. In recent years, intrusion detection method and key technology has become one of research focus in network security field.

IDS could be classified in different ways[24]. There are network based (NIDS)

and host based (HIDS) intrusion detection systems[185]. There are misuse-based intrusion detection and anomaly-based intrusion detection. It will described in detail in chapter 2.

Network Intrusion Detection Systems are placed at a strategic point or points within the network to monitor traffic to and from all devices on the network. A large NIDS server can be set up on a backbone network, to monitor all traffic, or smaller systems can be set up to monitor traffic for particular server, switch, gateway, or router. It can be shown in figure 1.3. IDS could be placed in (Local Area Network)LAN, (DeMilitarized Zone)DMZ or Internet area. Our research work is based on NIDS since the data we used is collected by a kind of NIDS.

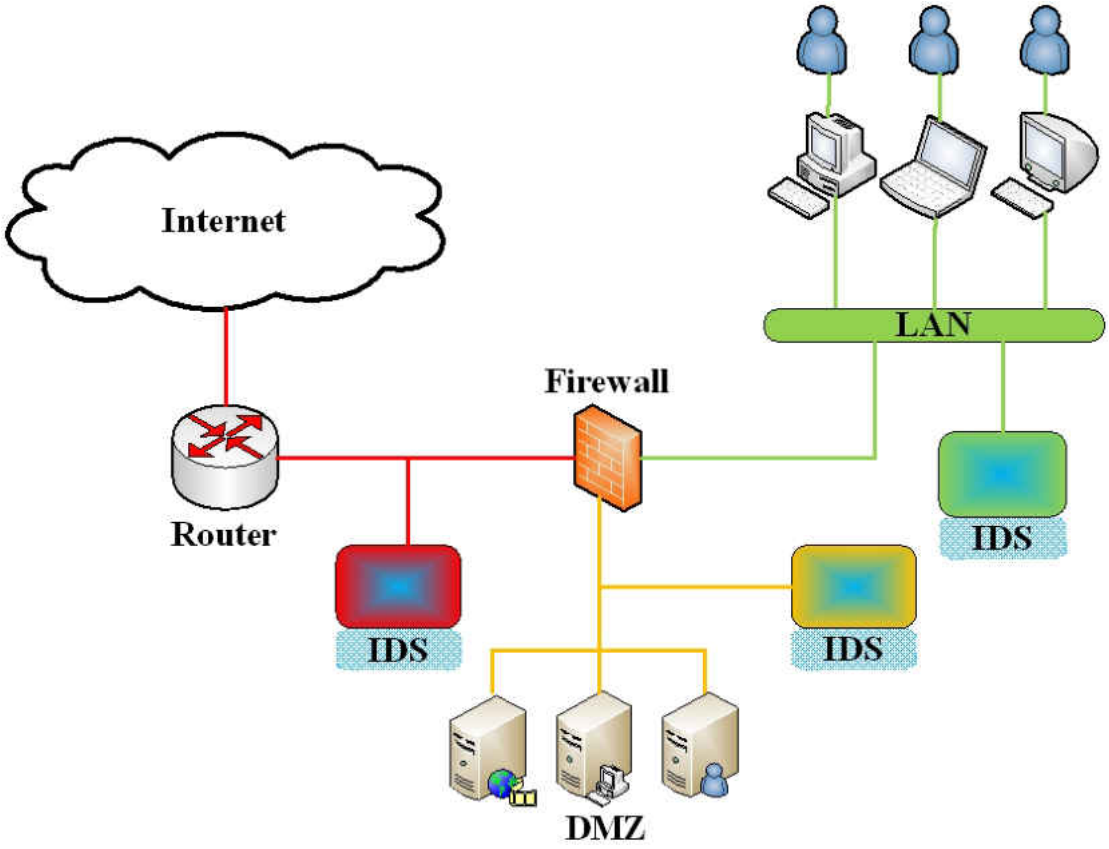

Figure 1.3: Computer network with intrusion detection systems

Traditional IDS is signature-based. It monitors packets on the network and compare them against a database of signatures or attributes from known malicious threats. And it could be seen as a classic kind of misuse-based IDS. The architecture of signature-based IDS is shown in figure 1.4. It has low false positives but could not

identify previously unknown attacks. The test dataset we used has unknown attacks and they will be introduced in detail in chapter 2.

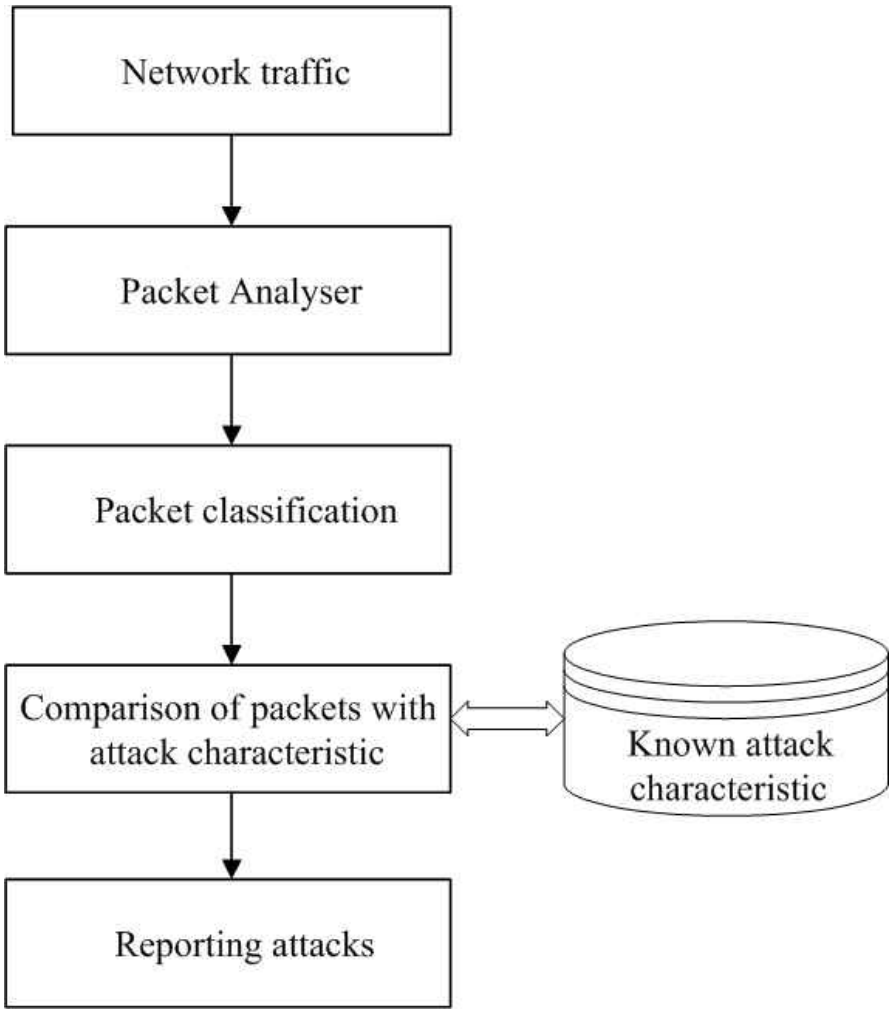

Figure 1.4: Signature-based IDS architecture

1.2.2 Feature Selection

Intrusion detection is a classification task, and it consists of building a predictive model which can identify attack instances. On the one hand, there are too many features or attributes which may contain false correlation, classification of anomaly intrusion detection systems is complex work. Moreover, many features may be irrelevant or redundant. For this reason, feature selection methods can be used to get rid of the irrelevant and redundant features without decreasing performance[52, 102, 12].

On the other hand, IDS usually runs day by day in real world. And the instance in IDS datasets are very huge and they take time to do classification or clustering. It will takes several days to get classification results from a dataset which has over 1 million instances. And if a dataset has a large number of instances and features, it will take large memory and computation resources to run. Thus, feature selection is very necessary to IDS datasets since they usually include a large number of instance and features.

1.2.3 Aims of the thesis

This thesis focuses on feature selection algorithms for intrusion detection system. And we test our algorithms by using some supervised methods, such as classification. Feature selection with classification method structure diagram is described in figure 1.5. Some clustering algorithms for IDS are presented as well since they can be seen as unsupervised methods. An algorithm to combine supervised and unsupervised methods is also introduced in this thesis. To test our algorithms work well or not, we choose a widely used dataset-KDD 99. Whether it will be introduced specifically in section 1.3.

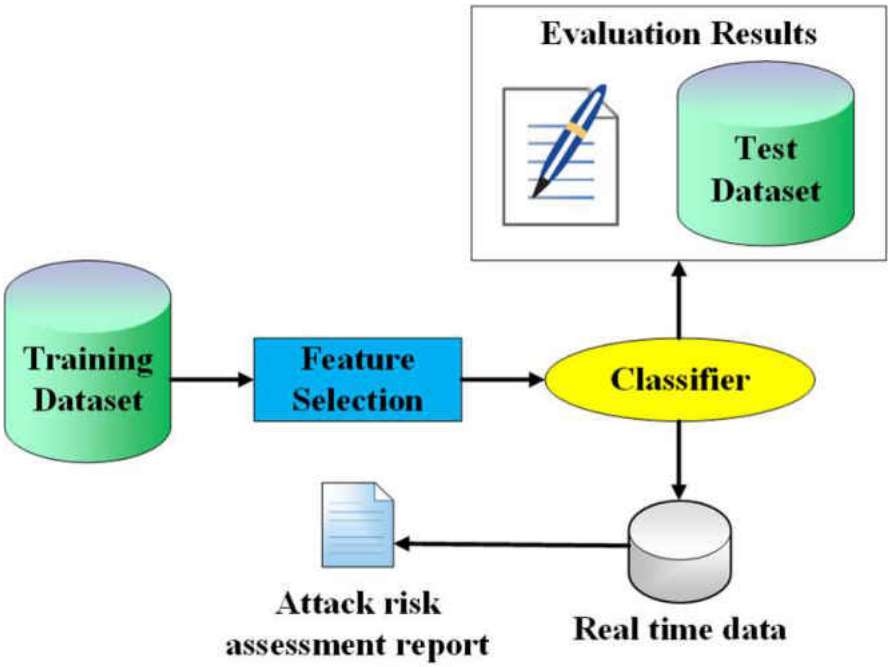

Figure 1.5: Framework of IDS based on feature selection

The motivation of the thesis is to find the relationship between features and class labels in a IDS dataset and design feature selection algorithms. Then, we select features from IDS datasets by using these feature selection algorithms. Moreover, we analysis the performance of selected features and compare with some other feature selection algorithms. At last, we test proposed algorithms on other datasets to see their effctiveness.

Besides dimensionality reduction and saving computation time, another advantage of feature selection is get rid of irrelevant and redundant features from datasets. A IDS dataset is usually got from network connections. Thus, it usually has many features and some of them are useless or counterproductive for classification. And these features can be seen as irrelevant and redundant. One of the purpose of feature selection is remove this kind of features.

1.3 KDD 99 dataset

Since 1999, SIGKDD (Special Interest Group on Knowledge Discovery and Data Mining) of ACM (Association for Computing Machinery) has been organizing an annual KDD (Data Mining and Knowledge Discovery) CUP competition. Later on, KDD 99 became the most popular dataset used for evaluation of anomaly detection methods. This dataset consists of the data generated from DARPA'98 IDS evaluation program, which comprises about 4 gigabytes of compressed raw (binary) tcpdump data of 7 week network traffic, and can be processed into around 5 million connection records with 100 bytes each. The test data of two weeks includes around 2 million connection records[100].

The definition of a connection is a TCP data packet sequence including data from source IP address to destination IP address in a predefined protocol (such as TCP or UDP) from beginning to end in a period of time. Each connection is classified as either attack or normal. An attack can be sub classified into four categories of 39 types. The 7 week training dataset only contains 22 types of attacks, and the test dataset includes other unknown 17 types[18]. It is notable that the probability distribution of the test data is not the same as the one of training data, and also that the test data contains certain attack types which do not appear in the training data. It is believed by some intrusion experts that most of the novel attacks are variants of known attacks, the signature of which is sufficient to capture novel variants[105, 153].

1.3.1 Dataset Description

The training dataset is composed by about 4,900,000 single connection vectors. Each of those vectors includes 41 features and one class label[80]. An attack can be classified into one of the four categories as below:

(1) Denial of Service Attack (DoS): Some computing or memory resources are made too busy or too full to accept legitimate requests, or to allow legitimate users to access a machine. E.g., ping-of-death, syn flood, smurf.

(2) User to Root Attack (U2R): An attacker gains access to a normal user account (perhaps by a dictionary attack, sniffing passwords or social engineering) and exploit the vulnerability in the system to gain root access to it. E.g., guessing passwords.

(3) Remote to Local Attack (R2L): By sending packets to a machine over a network, an illegitimate user exploits vulnerability of the machine to gain local access to it as a user. E.g., buffer overflow attacks.

(4) Probing Attack: In order to circumventing the security controls of a computer network, the attacker attempts to gather information of the network. E.g., port-scan, ping-sweep.

KDD 99 features are divided into four categories:

(1) Basic features: This category contains all the attributes extracted from a TCP/IP connection. The monitoring of these features will cause a fixed delay in detection.

(2) Content features: The features of suspicious behavior in the data portion should be captured in order to detect attacks. E.g. number of failed login attempts. Those features are called content features. The R2L and U2R attacks normally don't appear in intrusion frequent sequential patterns, as they have been embedded in the data portions of packets and only request a single connection. While the DoS and Probing attacks involve many connections to hosts and show the attribute of intrusion frequent sequential patterns.

(3) Time-based traffic features: Only the connections in the past 2 seconds are examined, which have the same destination host/service as the current connection, and of which the statistics related to protocol behavior, service, etc. are calculated.

(4) Connection-based traffic features: Some slow probing attacks scan the hosts/service at an internal much longer than 2 seconds, e.g. once in every minute, which cannot be detected by the time-based traffic features, as it only examines the connections in the past 2 seconds. In such case, the features of same destination host/service connections can be re-calculated at an interval of every 100 connections rather than a time window[50]

1.3.2 Statistical Observations

There are three components in KDD dataset, "10 % training KDD" dataset, "corrected KDD" dataset and "whole KDD" dataset. The "10 % training KDD" dataset is used for training purpose. There are 22 types of attacks in the training dataset. It is a more concise version of the "whole KDD" dataset, and with the attack types representing unequally, it includes more attack connections than normal. Also denial of service attacks account for the majority of the dataset[28]. The basic characteristics of KDD 99 training dataset are represented in figure 1.6 and 1.7.

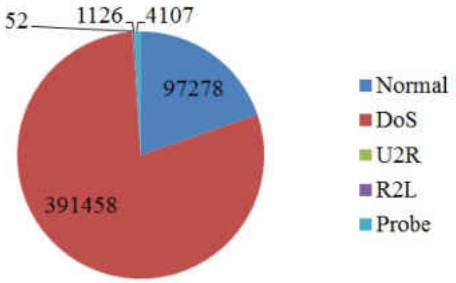

Figure 1.6: Basic characteristics of "10% KDD" training dataset

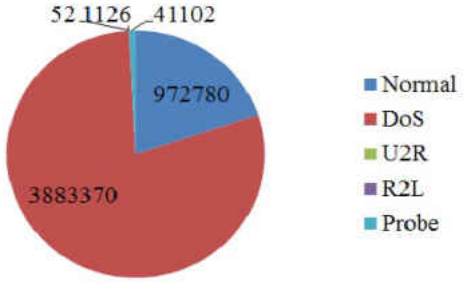

Figure 1.7: Basic characteristics of "whole KDD" training dataset

The "Corrected KDD" dataset is a test dataset with different statistical distributions from either "10% training KDD" or "Whole KDD". It contains 17 additional attacks[92]. The basic characteristics of "corrected KDD" dataset is shown in figure 1.8.

The list of class labels and their corresponding categories for "10% training KDD" and "Corrected KDD" dataset are described in table 1.1.

There are several categories of derived features. Basic features of individual TCP connections are shown in table 1.2. Content features are illustrated in table 1.3. Time-based traffic features only examine the connections in the past two seconds,

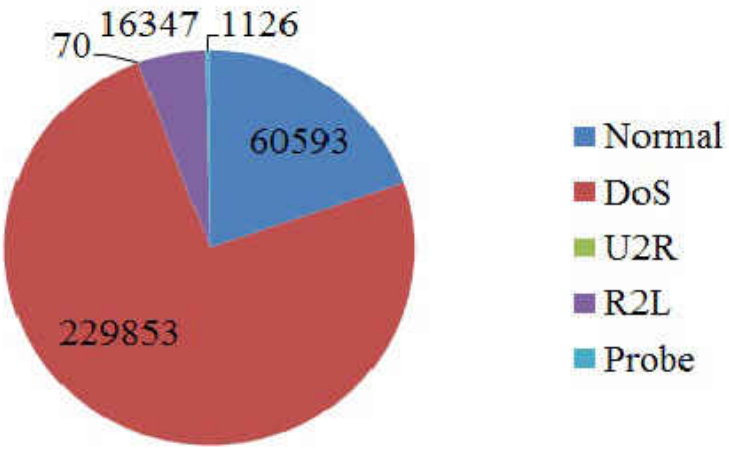

Figure 1.8: Basic characteristics of "corrected KDD" dataset

which have the same destination host/service as the current connection, and for which the statistics related to protocol behavior, service, etc are calculated. The "same host" and "same service" features are together called time-based traffic features, which are illustrated in table 1.4[167].

There are probing attacks which scan the hosts/ports at a much longer internal than two seconds, e.g. once in every minute. Such connections are not able to be detected by time-based traffic features. Therefore, connection-based traffic features[181] are introduced to examine the connection records at a window of 100 connections for the connections to the same host/service. It is shown in table 1.5.

1.4 Structure of Thesis

This section outlines the structure of the remainder of this thesis. Figure 1.9 illustrates the relationships between the individual chapters (other than the introduction). The direct dependencies between the chapters are denoted using solid arrows, where conceptual linkages are symbolised using dashed lines. The contributions of this thesis are mainly shown in chapter 3 to chapter 6 and we design and implement six algorithms. Four of the proposed algorithms are feature selection algorithms and they could help for classification. Another two algorithms could help for clustering and conbine with feature selection.

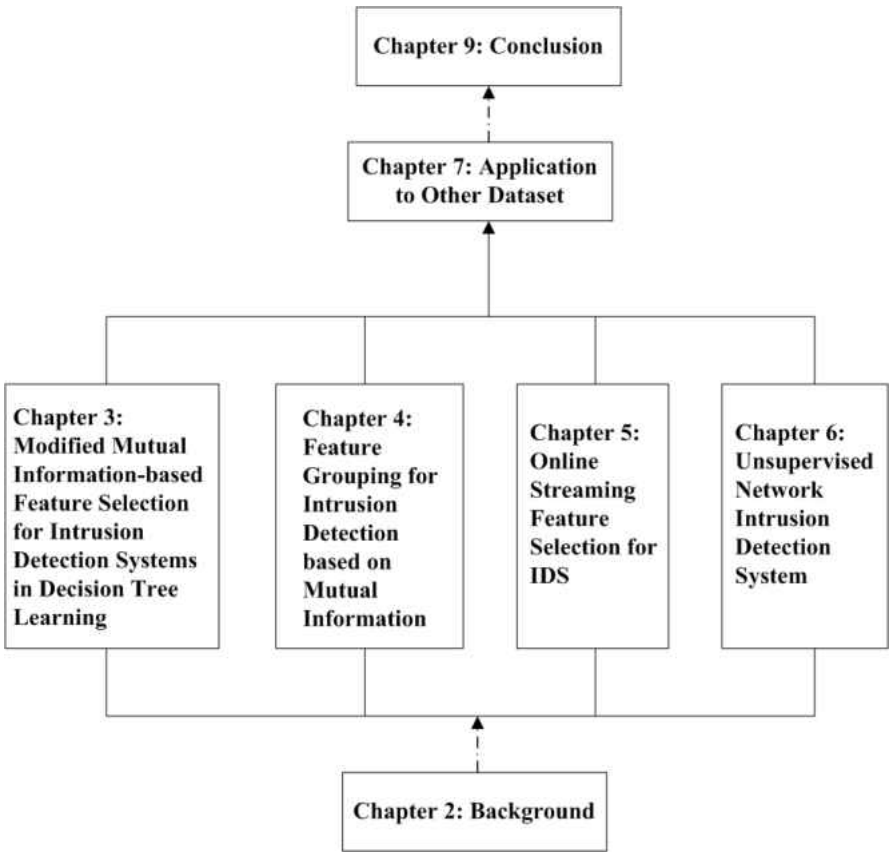

Figure 1.9: Relationships between thesis chapters

Chapter 2: Background

This chapter provides a background introduction to intrusion detection system and feature selection. This chapter also provides a comprehensive review of the most recent methods for feature selection.

Chapter 3: Modified Mutual Information-based Feature Selection for Intrusion Detection Systems in Decision Tree Learning

Mutual information-based feature selection method was first proposed by Battiti in 1994. It was modified by Huawen Liu in 2009 and by Fatemeh in 2011. This chapter proposes a modified mutual information feature selection method based on Battiti's work and compares the resulting performance with Huawen's work. After we calculate the selected features, we use the decision tree classification methods to evaluate the performance.

Chapter 4: Feature Grouping for Intrusion Detection based on Mutual Information

This chapter presents two feature grouping methods for the selection of features for intrusion detection. One method is based on mutual information theory and is tested against KDD CUP 99 dataset. It ranks the mutual information between features and uses the fuzzy C means algorithm to compose groups. Another one is based on agglomerative hierarchical clustering method and is tested against KDD CUP 99 dataset as well. Agglomerative hierarchical clustering method is used to construct a hierarchical tree and it is combined with mutual information theory. Groups are created from the hierarchical tree by a given number.

Both of these two algorithms use the same selecting strategy of feature grouping. The largest mutual information between each feature and a class label within a certain group is then selected. The performance evaluation results show that better classification performance can be attained from such selected features.

Chapter 5: Online Streaming Feature Selection for IDS

Unlike the existing studies on feature selection, online feature selection aims to solve the feature selection problem by online learning approach. Streaming features are features in dataset which flow one by one over time without changing the number of training samples. In this chapter, I introduced two online feature selection algorithms. One analyses relevance and redundancy between features and labels or features. And users need to input two thresholds to help the algorithm judging. Another algorithm design a criterion by relevance and redundancy between features and labels or features. And users need to input a desired number of features will be selected. From the comparison with other feature selection algorithms we proposed before, we could see that OSFS algorithms could get better performance.

Chapter 6: Unsupervised Network Intrusion Detection System

This chapters introduces two algorithms. One is cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm. This method for classification is proposed consisting of a combination of feature selection, normalization, fuzzy C means clustering algorithm and C4.5 decision tree algorithm. The aim of this method is to improve the performance of the classifier by using selected features. The fuzzy C

means clustering method is used to partition the training instances into clusters. On each cluster, we build a decision tree using C4.5 algorithm. Experiments on the KDD CUP 99 data set shows that our proposed method in detecting intrusion achieves better performance while reducing the relevant features by more than 80%.

Another one is new evidence accumulation clustering with hierarchical clustering algorithm. It deals with features of the dataset one by one. In other words, it clusters a feature one by one rather than clustering all features at the same time. The algorithm uses voting mechanism to compose a co-association matrix and uses hierarchical clustering algorithm with single link to get the final partition.

Chapter 7: Application to Other Datasets

This chapter will test all algorithms proposed in chapter 3, 4, 5, 6 on other datasets and other areas, and compare performances with other methods or algorithms.

Chapter 8: Conclusion

This chapter summarises the key contributions made by the thesis, together with a discussion of topics which form the basis for future research. Both immediately achievable tasks and long-term projects are considered.

Table 1.1: Class labels details that appears in "10% training KDD" and "Corrected KDD" dataset

Category	Training Data		Test Data	
	Class labels(23)	Number	Class labels(38)	Number
Normal	normal	97278	normal	60593
Probe	ipsweep	1247	ipsweep	306
	nmap	231	nmap	84
	portsweep	1040	portsweep	354
	satan	1589	satan	1633
			saint	736
			mscan	1053
DoS	back	2203	back	1098
	land	21	land	9
	neptune	107201	neptune	58001
	pod	264	pod	87
	smurf	280790	smurf	164091
	teardrop	979	teardrop	12
			apache2	794
			mailbomb	5000
			udpstorm	2
			processtable	759
U2R	perl	3	perl	2
	rootkit	10	rootkit	13
	loadmodule	9	loadmodule	2
	buffer_overflow	30	buffer_overflow	22
			httptunnel	158
			ps	16
			sqlattack	2
			xterm	13
R2L	ftp-write	8	ftp-write	3
	guess_passwd	53	guess_passwd	4367
	multihop	7	multihop	18
	phf	4	phf	2
	imap	12	imap	1
	spy	2	spy	
	warezclient	1020	warezclient	
	warezmaster	20	warezmaster	1602
			named	17
			xsnoop	4
			xlock	9
			sendmail	17
			worm	2
			snmpgetattack	7741
			snmpguess	2406
total	494021		311029	

Table 1.2: Basic features of individual TCP connections

feature name	description	type
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
service	network service on the destination, e.g., http, telnet, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of "wrong" fragments	continuous
urgent	number of urgent packets	continuous

Table 1.3: Content features within a connection suggested by domain knowledge

feature name	description	type
hot	number of "hot" indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of "compromised" conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if "su root" command attempted; 0 otherwise	discrete
num_root	number of "root" accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	1 if the login belongs to the "hot" list; 0 otherwise	discrete
is_guest_login	1 if the login is a "gues" login; 0 otherwise	discrete

Table 1.4: Time-based traffic features computed using a two-second time window

feature name	description	type
count	number of connections to the same host as the current connection in the past two seconds Note: The following features refer to these same-host connections.	continuous
serror_rate	% of connections that have "SYN" errors	continuous
rerror_rate	% of connections that have "REJ" errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds Note: The following features refer to these same-service connections.	continuous
srv_serror_rate	% of connections that have "SYN" errors	continuous
srv_rerror_rate	% of connections that have "REJ" errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous

Table 1.5: Connection-based traffic features

feature name	description	type
dst_host_count	count of connections having the same destination host	continuous
dst_host_srv_count	count of connections having the same destination host and using the same service	continuous
dst_host_same_srv_rate	% of connections having the same destination host and using the same service	continuous
dst_host_diff_srv_rate	% of different services on the current host	continuous
dst_host_same_src_port_rate	% of connections to the current host having the same src port	continuous
dst_host_srv_diff_host_rate	% of connections to the same service coming from different hosts	continuous
dst_host_serror_rate	% of connections to the current host that have an S0 error	continuous
dst_host_srv_serror_rate	% of connections to the current host and specified service that have an S0 error	continuous
dst_host_rerror_rate	% of connections to the current host that have an RST error	continuous
dst_host_srv_error_rate	% of connections to the current host and specified service that have an RST error	continuous

Chapter 2

Background

THE detection of network attacks is a important task for network operators in today's Internet. The principal challenge in automatically detecting network attacks is that these are a moving and ever-growing target[58, 169]. Intrusion detection systems usually detect anomaly attacks by monitoring packets. We often use machine learning technologies to identify whether traffic data is normal or anomalous[54]. Two common machine learning methods are classification-based and clustering-based. But some of the methods lose effectiveness or even become invalid in this area since data volume is often very large. Moreover, traffic data for the network contains many features, and some of the features are irrelevant or redundant. Thus, we usually use feature selection algorithm to remove irrelevant and redundant features. This chapter will introduce intrusion detection system, feature selection, and machine learning methods.

2.1 Intrusion Detection System

Intrusion detection system (IDS) is used to moniter malicious events on computers or networks. IDS could discover attack indications and illegal actions which break security policy in systems or networks. A good IDS can not only help a network administrator understand attacks occuring in a network system at any time, but also provide an important basis to constitute network security defence policy[40, 150].

Research of intrusion detection technologies come from 1980s. Since Denning proposed the first IDS model in 1985, more and more researchers have done lots of

work on how to construct an effective IDS model. Since IDS needs to detect, defend against and respond to computer attacks, researchers have to consider many problems when they construct IDS models. Such as data collection, intrusion identifying, reporting and response. The structure of IDS construction is shown in figure 2.1. IDS is composed by four parts as follows[118, 131].

1. Monitoring object. It is monitored and it can be a host or a network.
2. Data collection and storage. This part collects all data from every event, and converts the data to a proper format to store.
3. Data analysis and management. This is a core part in IDS. It searches suspected actions and generates a signal when it detects an attack. Then, IDS deals with the attack or send a signal to network administrator to handle.
4. Signal. It can be seen as an output of IDS. The output is an automatic response or an alarm to network adminstrator.

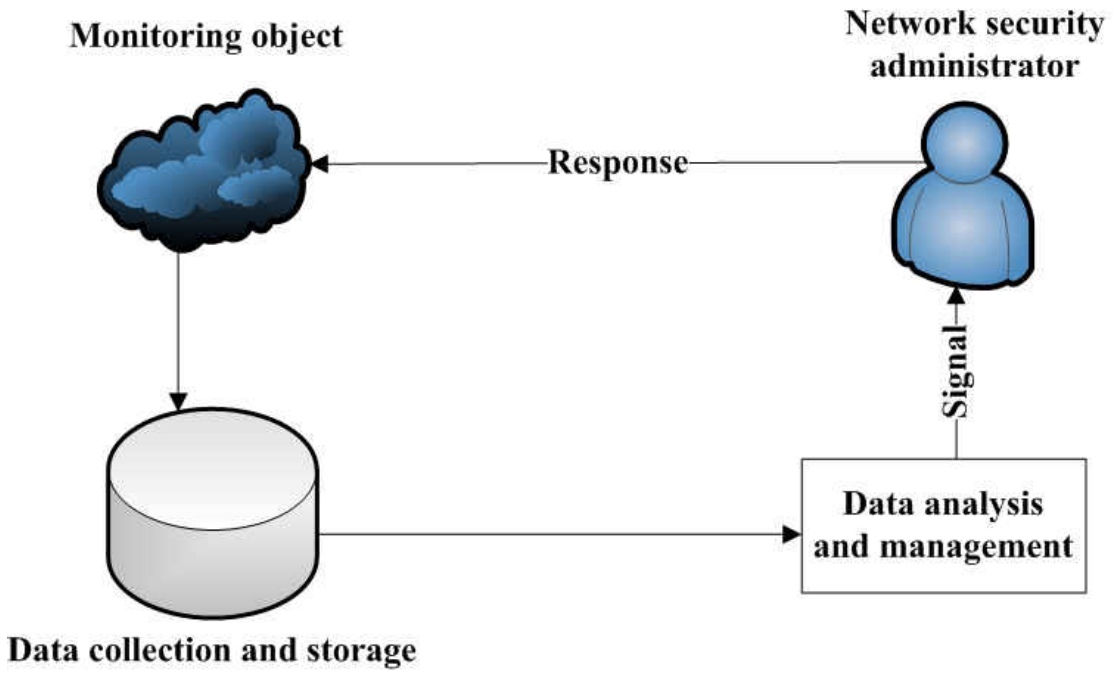

Figure 2.1: IDS structure

2.1.1 Brief history of IDS

In 1980, Anderson J P. put forward the concept of an intrusion detection. He defines intrusion attempt or threat as an attempt caused the system to be unavailable or unreliable by unauthorized access information or operating information[4]. Anderson J P. proposed intrusion detection idea according to audit record of operating system. But researchers has paid little attention to this idea, focusing instead on encryption and denial of access to the data from an authenticated host[86]. In 1985, Denning D E. proposed a IDS model which is called Intrusion Detection Expert System (IDES). And it is composed by host, object, audit record, profile characteristic, anomaly record and activity rules[39]. This model is independant from system platform, application enviroment, system weakness and types of attack. And the model provides a general framework for IDS. In 1988, Teresa L improved the model and created a real time IDS that detected attacks as data was received[119]. And Teresa's model is used to detect intrusions behavior for a single host. In 1990, Heberlein L T presented network-based intrusion detection, and proposed Network Security Monitor (NSM) which detects suspicious behavior by monitoring network data in local area network, rather than checking audit record in host.

From 1990s, the research on intrusion detection is increasing gradually. Some American research institutions combined host-based IDS with NIDS together, design a distributed IDS[186]. Crosbie M and Spafford G use autonomous agents to improve the scalability, maintainability, efficiency and fault tolerance of intrusion detection system[34]. Sandeep Kumar studied IDS based on immune principle[151, 155]. Anderson R introduced the information retrieval technology into the field of intrusion detection[6]. Lane T studied anomaly detection of user's behavior based on machine learning[95, 96]. Lee W applied data mining approaches on IDS[179].

2.1.2 Classification of IDS

Based on the four parts of IDS in figure 2.1, we could divide IDS into different categories. According to difference of data collection and storage, IDS be classified to Host-based IDS (HIDS), application-based and Network Intrusion Detection System (NIDS). It is shown in figure 2.2. NIDS takes raw packets in network as data source. Sensors collects packets from a protected network to determine whether the network is normal or not. Response module will alarm to administrator when sensors detect an attack[107, 128]. A HIDS emerges from 1980s, and the network has not developed

as common and complex as today's. Thus, it usually monitors packet based on a host and compare packets' behaviour with signature. And it will alerts users when an attack is detected. Sensors of host-based IDS get history audit data from host operating system. Application-based IDS run on individual host as well. And sensors of application-based IDS gain log files from uses' application and software[25, 43].

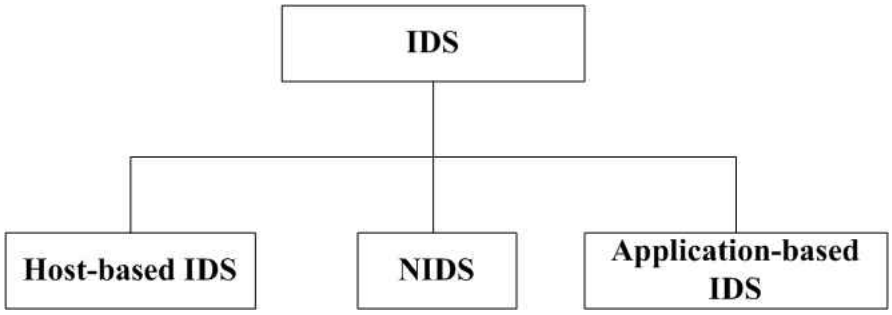

Figure 2.2: IDS classification based on data collection and storage

On the basis of difference of data analysis and process unit, IDS can be separated into misuse detection and anomaly detection. It is shown in figure 2.3. Misuse detection is used to analyze and detect intrusion. This method generally takes intrusion behavior as a pattern or a character. And it establishes a intrusion mode characteristic database based on known intrusions behavior patterns. The detection will be monitoring system or the user's actual behavior patterns and match them with the database. According to the results of the matching, the system will determine whether there is a intrusion [82]. Supervised machine-learning methods could help to compose signatures. Misuse detection systems are highly effective to detect those attacks which they are programmed to alert on. However, they cannot detect new attacks, since they cannot recognize those attacks which do not match their lists of signatures[145, 77]. Misuse detection based intrusion systems can be divide into stateless and stateful. Stateless misuse detection systems use only existing signature. However, stateful misuse detection systems use not only existing signatures, but also previous signatures[140]. On the contrary, anomaly detection will create a normal operation model for users. Any operation does not comply with the normal behavior will be prevented. Anomaly detection principle is take every exception as a possible attack. Thus, this detection method can detect unknown attacks[22]. Anomaly detection based intrusion system can also be further classified into self-learning and rule-based. The difference is that rule-based intrusion detection system will be fully specified in advance of normal rules. But self-learning systems typically need to have a training process, which can let the system know what is normal network behavior.

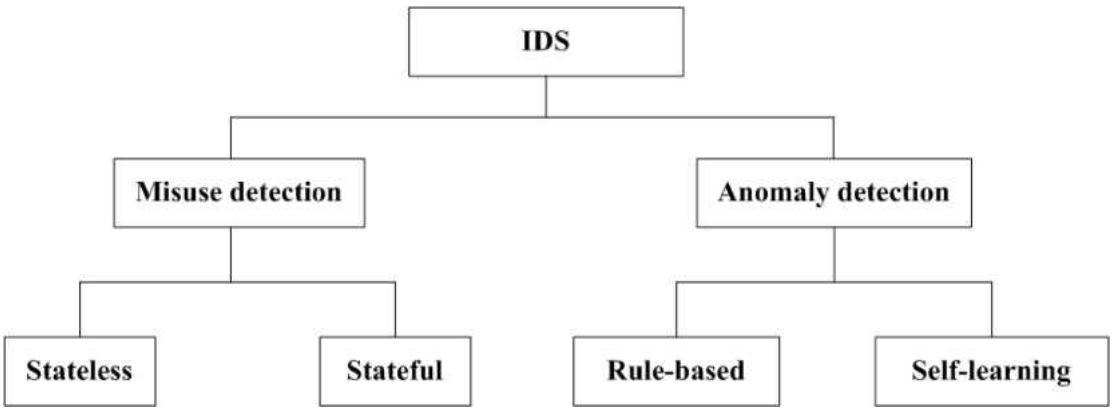

Figure 2.3: IDS classification based on data analysis and process

In accordance with the reponse mechanism, IDS falls into reactive response IDS and passive response IDS. And according to usage frequency, IDS can be divided into online IDS and offline IDS.

2.1.3 Approaches to Intrusion Detection

As it is shown in section 2.1.2, intrusion detection can be classified in several different ways. One way commonly used is according to data analysis and process unit. As stated above, misuse detection uses a abnormal behavior rule database to distinguish whether an action is an attack. Any action consistent with the rules in the database will be prevented. Rule also named as signature and is constructed by malicious pattern. Misuse detection has a strong detection ability, its disadvantage is the need to update the pattern database, and it is difficult to detect unknown intrusion. The core of the misuse detection is how to express the intrusion behavior, and how to make the intrusion actions to ensure its completeness. As new types of attack and network vulnerabilities occur constantly, it is impossible to keep the pattern database reflecting the potential attacks. This is main reason to affect false negative rate.The current misuse detection mainly has three methods as follows.

1. Simple pattern matching

   Simple pattern matching is the most common method of misuse detection, which has the advantages of being easy to implement, with high detection efficiency, and strong real-time performance, but can only be applied to a relatively simple attack mode, and false positive rate is high[42, 136]. The

well-known network intrusion detection tool Snort uses the simple pattern matching method. It uses rule base to describe the intrusion behavior that has been known and the rule base uses the text file to store. It has good readability and can be modified[170, 15].

2. Expert system

   Expert system is one of the earliest misuse detection schemes, and has been adopted by many classical intrusion detection models. When the expert system works, user have to input the information of the known intrusion behavior to the expert system in the form special format which is expert system required. Expert system constructs a rule base by using these information. The expert system matches the corresponding observation events and rules in the rule base to determine whether the intrusion occurs. For users, the expert system is an autonomous "black box", users do not need to understand or interfere with their internal reasoning and decision-making process[62, 33]. The main problems existing in the expert system are the maintenance of the rule base is complex, and we need to consider the relationship between the rules when changing the rules. And another problem is the low efficiency in dealing with massive data[14, 70].

3. State transition diagram

   State transition analysis using state transition diagrams to represent and detect known intrusions. In a state transition diagram, an intrusion behavior is represented as a series of state transfer processes, and the process starts from an initial state until the final state is invaded. The advantages of state transition analysis are as follows[149]. First, it does not depend directly on the detailed data, but identifies the key features of intrusion activities that need to be detected. Second, it can be detected before the invasion has been completed, so as to facilitate the timely response measures to prevent the invasion. Third, it can detect slow attack and cooperative attack[177, 184]. The weakness of state transition analysis are as follows. First, since the state transition diagram can describe the intrusion behavior which have to cause the obvious system state change, many intrusion behaviors can not be described by the state transition diagram. Second, intrusion behavior is simply described as a state sequence, some of the more complex behavior can not be described, such as concurrency, conditions, etc.. Third, the state transition diagram need to determine whether

the system behavior meets the requirements of the intrusion proposition before it is checked and matched[68].

Anomaly detection will judges a intrusion when there is a certain difference between monitored system or the user's actual behavior and normal behavior[103, 176]. The advantage of anomaly detection is that there is no need to have much knowledge about system defects, and has strong adaptability, which can detect unknown intrusions or new intrusion patterns. The core problem of anomaly detection is how to represent the normal behavior of the system or the user. The current anomaly detection mainly has three methods as follows[98, 91].

1. Statistical method

   Anomaly detection based on statistical method is the use of specific statistical model of the system or the user normal behavior for learning. And it identifies abnormal behavior which is a deviation behavior compare with normal behavior based on large statistical data. The key of statistical method is the selection of statistical object and statistical model, and the training of statistical model. The following are some of the possible statistical object[78].

   a) User login and activity

      User login frequency, activity duration, password error number, etc..

   b) Command and program execution

      Command execution frequency, the use of the program running, etc.

   c) The operation of the file

      File read, write, create, delete frequency and the failure frequency of these operations, etc..

   This method is not very difficult to select the appropriate statistical model for the specific intrusion detection. And it can be seen as its advantage.The weakness is the threshold value is difficult to determine, too large or too small value will affect the accuracy of detection. Moreover, many of the system or user behavior is very difficult to use simple statistical model to describe, and the complex statistical model requires high calculation[120].

2. Anomaly detection based on Immune Principle

   Biological immune system against pathogens or non detection of the organization itself is quite precise and recognition of "self/nonself" is one of the most basic and most important functions[53]. The researchers in the United States have found a certain similarity between the biological immune system and the computer system security protection mechanism. This method using system calls as audit data, through the fixed length sequences of system calls to describe process privilege of normal behavior profile and sequence between the "distance" as reflected in the differences between the patterns of behavior measure[117]. Experiments show that the anomaly detection system based on immune principle can detect many attacks that exploit the vulnerability of the program. However, this detection method has a limitation that it could only detect the attacks who use the privilege of the process [56].

3. Artificial neural network

   Artificial neural network is a kind of artificial intelligence method, which is based on the understanding of the structure and operation mechanism of human brain. Artificial neural network model is based on the mathematical model of neuron. Artificial neural network models are represented by neural networks, network topology and learning rules[135]. Artificial neural network has the ability of massively parallel processing and distributed information storage, good adaptive, self-organizing, and strong learning function, associative function and fault tolerance function[137]. When the traditional method is unable to solve or the effect is not good or when the characteristics of the original data is not understood or can not be described by a mathematical model, the neural network can show the superiority[17].

   Artificial neural network has two ways to detect anomalies. One method use a large number of instances to train neural network and it gains knowledge of normal behaviour. And anomaly behaviours could be detected by comparison with knowledge. Another method is to train neural network by users' representative command sequence. And then the network could create characteristics table for users. The network's prediction error rate for next event can measure the abnormal degree of user behaviour.

2.1.4 Challenges in Intrusion Detection

Intrusion detection will develop towards distributed, intelligent, high detection speed, high accuracy and high security. And the research focus of intrusion detection will include the following.

1. Distributed intrusion detection

   Distributed intrusion detection system is mainly for large networks and heterogeneous system, which uses distributed structure, collaborative processing and analysis of a variety of information, and a single architecture of intrusion detection system compared with greater detection ability[76].

2. Intelligent intrusion detection

   Intelligent intrusion detection method is the present stage, including machine learning, neural networks, data mining, and other methods. It has carried out various intelligent techniques in the application and research of intrusion detection. The main purpose of the study is reduced detection system false alarm and false alarm probability, improve the system self learning ability and real-time response. From the current research results, the intrusion detection method based on intelligent technology has many advantages, and has good development potential[20].

3. Intrusion detection based on protocol analysis

   The calculation amount of intrusion detection based on protocol analysis is relatively small. It can be used to detect the presence of a high degree of regularity of network protocol, even in high load network, it is not easy to generate packet loss[11].

4. Combined with operating system

   Closely integrated with the operating system can enhance the intrusion detection system to new attack detection capabilities.

5. Application layer intrusion detection

   The semantics of many intrusions can be understood only in the application layer, and the detection of this kind of intrusion needs to be realized by analyzing the application layer[192].

6. High speed packet capture technology

   For network intrusion detection system, high-speed packet capture can reduce the resource consumption and improve the detection speed.

7. Efficient pattern matching algorithm

   As intrusions become more diverse and complex, more and more complex models need to be stored in rule base. And complexity of intrusion model definition are higher and higher. Therefore, it is urgent to research and use efficient pattern matching algorithm[178].

8. Test and evaluation of intrusion detection system

   The establishment of common intrusion detection system evaluation method and testing platform, which is very important to promote the application and popularization of intrusion detection system, has become another important direction of intrusion detection research[65].

9. Standardization of intrusion detection system.

   There is no formal international standards of intrusion detection system so far. And it is not conducive to the development and application of intrusion detection system.

10. The interaction between intrusion detection system and intrusion detection system and other security components.

    Intrusion detection system could combine with other IDS or security components by cascaded connection or integration.

11. Research on the security of intrusion detection system itself.

    Intrusion detection system has its own security problem as well. And there should be research on how to protect itself against network attacks.

2.2 Feature Selection

Feature selection is defined as follows: given a set of candidate features, select a subset or a feature that performs the best under some classification algorithms. This process can reduce not only the cost of recognition by reducing the number of features, but also provide a better classification accuracy due to finite dataset size

effects[71]. From the definition of feature selection, we could see evaluation criteria are very important when given a specific learning algorithm and a dataset[143, 83]. Generally, a feature selection algorithm includes four parts which are shown in figure 2.4, generation, evaluation, stopping criterion and validation. Feature selection process is can be seen as removing irrelevant and redundant features. Irrelevant means features have little correlation with class labels. And redundant features have strong relationship with selected features. Thus, both irrelevant and redundant features are no help for classification.

The candidate subset generation is a process of searching feature subsets, and the obtained subset will be used as input for evaluation function. The selection of the initial subset is the start of the feature selection algorithm, and the starting point of subset generation process, which is divided into three categories. (1) Initial subset is empty. In the process of searching, the algorithm adds candidate features to candidate subset one by one. This method is called forward search. (2) Initial subset is the same as feature set of a given dataset. And it excludes irrelevant or redundant features from the initial subset step by step in the search process, namely the backward search. (3) The initial subset is generated randomly, then the feature is added or deleted one by one in the search process[97, 161].

The evaluation function is used to evaluate the merits of the candidate subset obtained by the search. It will compare evaluation value with the best optimal value stored before. If the evaluation value is higher, the primary candidate subset will be replaced[168, 114].

Appropriate termination conditions can avoid exhaustive or infinite loop state in feature search procedure. The subset search be applied to strategy and evaluation function is an important factor that influences the selection of termination condition. The best feature subset search strategy can improve the speed of feature selection to find the optimal solution[160]. And better evaluation function can ensure that the selected feature subset has higher classification distinguishing ability, and improve the performance of the algorithm. The termination condition based on search strategy can be feature number achieve the specified threshold or the iterations number of search achieve the specified threshold. Termination criteria based on evaluation function can be optimal solution has been found or could not obtain a higher evaluation value by increase or decrease the number of feature subset[108].

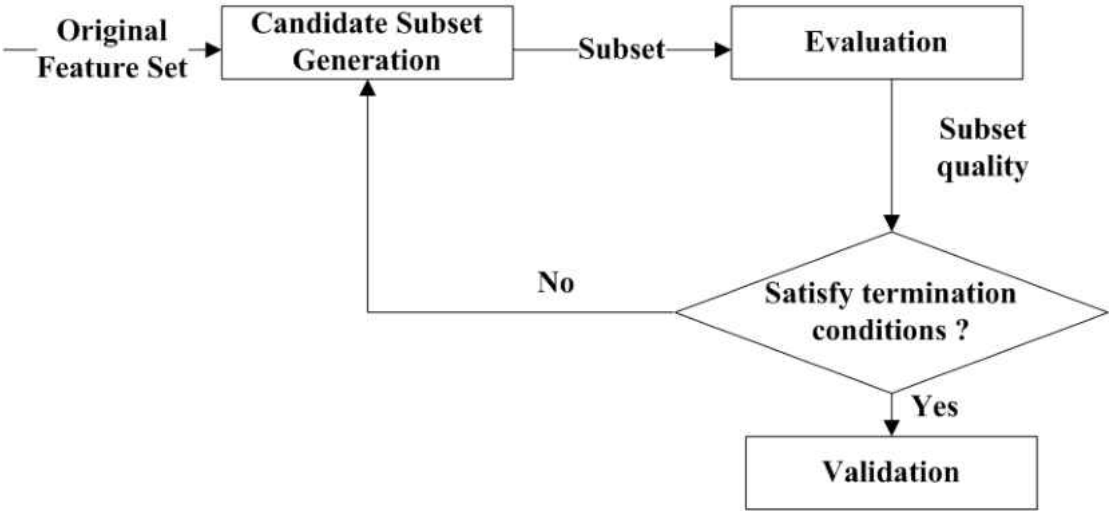

Figure 2.4: Feature selection process

Validation part verifies the the classification effectiveness performance of the feature selection results in certain conditions. It is not a part of feature selection process but is necessary in the practical application. Validation is usually to train and test feature subset in some kind of classifier and compare prediction results with original dataset results, or other feature selection results[183]. Comparison may be classification accuracy or computational complexity and so on.

2.2.1 Feature Selection evaluation measure

There are four main models dealing with feature selection: wrapper methods, filter methods, hybrid methods and embedded methods. It is shown in figure 2.5. In the embedded model, feature selection is integrated into the process of training for given methods. For example, some decision tree algorithms like ID3, C4.5 or Breiman's CART algorithm. These algorithms choose the best feature which is good for classification in each node. And then they split sub-space based on selected feature. The algorithms repeat this process until termination condition is reached[19, 74]. Embedded methods attempt to find an optimal subset of features in the process of model building. These methods depend directly on the nature of the classification method used[55]. In general, embedded methods present important advantages in terms of variable and model interaction, capturing accurately the dependencies between variables, being computationally less demanding than wrapper methods. However, these techniques are conceptually more complex, and modifications to the classification algorithm may lead to a poor performance[122].

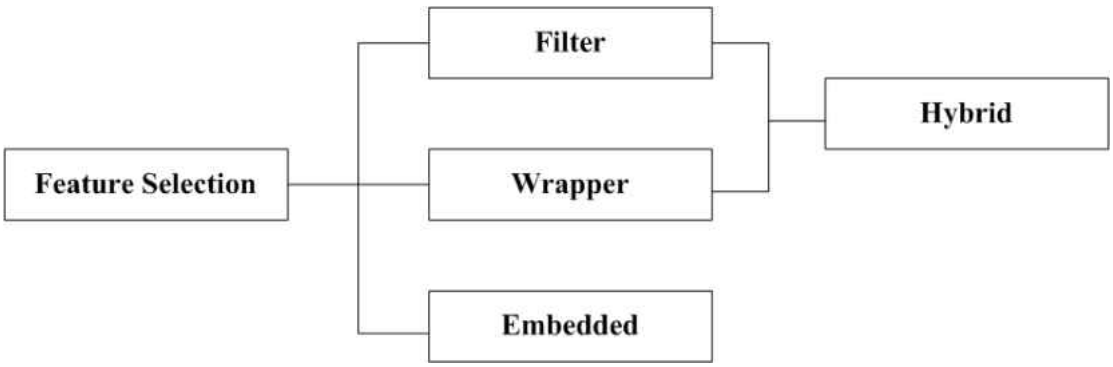

Figure 2.5: Types of feature selection evaluation measure

Filter methods' evaluation criteria are independent of learning algorithms and they mainly identify a feature subset from the original space on the basis of given evaluation criteria. It is described by figure 2.6. The evaluation criteria depends on datasets and filter methods usually select a feature or subset who could achieve highest degree relate to objective function. And it is generally considered that selected feature or subset has higher accuracy for learning algorithms. There are many evaluation methods for filter, such as inconsistency, correlation, information gain and so on [189, 57]. Filter methods have low computational complexity, high efficiency, strong versatility, are suitable for the large-scale data[94, 122].

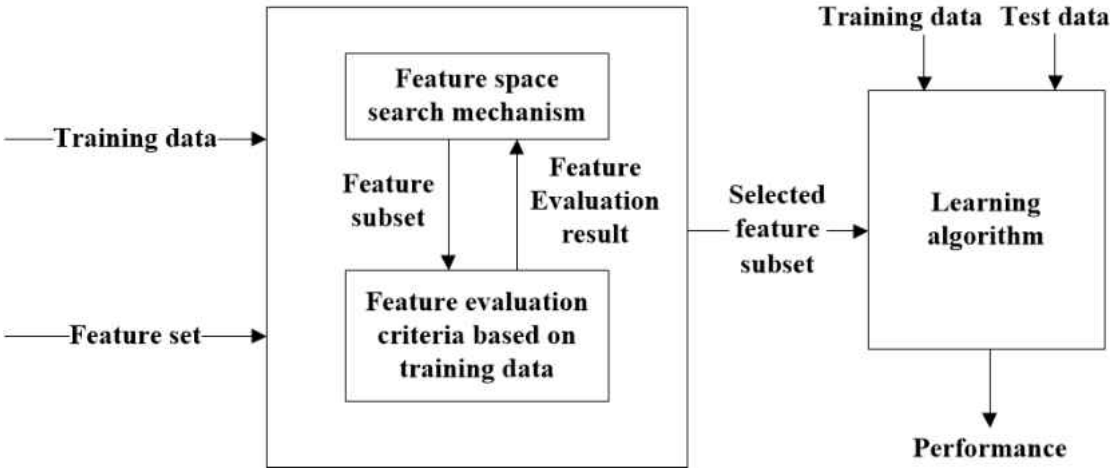

Figure 2.6: Filter-based feature selection flow

Wrapper methods was first proposed by John in 1994 and it is shown in figure 2.7[75]. Wrapper methods optimize a classifier as part of the selection process and choose those features with high prediction performance induced by specified learning

algorithms [121, 154]. Selected feature subset will vary depending on different learning algorithms. Therefore, the best evaluation criteria is the performance of learning algorithm which is used on selected feature subset. Wrapper methods have no limitation to learning algorithms. And decision tree, KNN, bayesian network and SVM could all be used for wrapper methods[57, 84]. In general, wrapper methods could get better subsets than filter methods. But they take long processing times, have low adaptability and need to train for different learning algorithms.

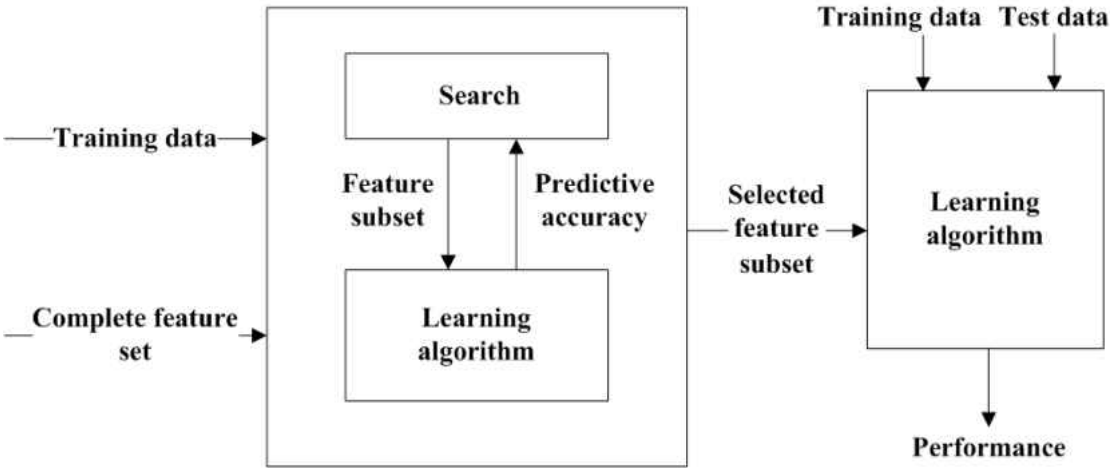

Figure 2.7: Wrapper-based feature selection flow

Hybrid methods combine filter method and wrapper method and take advantage of both of them [101, 134]. The hybrid mechanism is typically by two steps. At first, candidate features are preprocessed by filter methods and irrelevant features are removed. Thus, the dimension of dataset could be reduced. Then, Hybrid methods select features by wrapper methods and classification learning algorithm is used to evaluate the selected subsets[63, 195].

2.2.2 Feature Selection Approaches

In this section, some feature selection approaches are introduced. They are used to compare with proposed algorithms in the following chapters.

2.2.2.1 DMIFS

Feature selection using dynamic mutual information (DMIFS) was proposed by Huawen Liu in 2009[115]. As it is shown in algorithm 2.2.1, T denotes training

dataset and it is described as $D(F, C)$. C represents class labels. And S and F are selected and candidate feature subsets, respectively. DMIFS uses semi-supervised learning method which combine supervised and unsupervised methods. And semi-supervised learning takes advantage of labeled instances and unlabeled instances to do training and classification.

Normally, we can divide the instances in $T = D(F, C)$ into two types: labeled and unlabeled. We set the stopping condition as that when the selected features have the same information as the original features, the selection procedure will cease, which is frequently used in feature selection. When there are still unlabeled instances in D, the procedure will continue and pick out the candidate features from F. Assume that S is the subset of selected features, and the instances D are classified into two categories according to the labels C. D_u and D_l are unlabeled and labeled instances respectively.

Algorithm 2.2.1: Feature selection using dynamic mutual information

Input: A training dataset $T = D(F, C)$
Output: Selected features S

```
Initialize relative parameters: F← 'initial set of all features', C ← 'class labels',
S = ∅, D_u = D, D_l = ∅ ;
repeat
    for each feature f ∈ F do
        Calculate its mutual information I(C; f) on D_u ;
        if I(C; f) = 0 then
            F = F − {f}
    Choose the feature f with highest I(C; f) ;
    S = S ∪ {f} F = F \ {f} ;
    Obtain new labeled instances D_l from D_u induced by f ;
    Remove them from D_u, D_u = D_u \ D_l ;
until F = ∅ or D_u = I_T;
Return Selected features: S.
```

The algorithm estimates mutual information for each candidate feature in F with the label C. When the mutual information of feature is zero, the feature will be eliminated from F during the calculation. So the probability distribution of the feature is completely random, and the feature will not be used in the prediction of the unlabeled instances D_u, to make sure that the feature with the highest mutual information will be selected. In order to ensure that the selected feature will not be

re-calculated in the estimation of mutual information in the next round, the selected feature will be kept aside and later discarded from D_u, since new labeled instances D_l will be produced from D_u. After that, the algorithm runs into next round and picks up other candidate features. The procedure will continue until no candidate features in F or the number of the unlabeled instances is equal to inconsistency count of T.

2.2.2.2 mRMR

Maximum relevance minimum redundancy (mRMR) was proposed by Hanchuan in 2005[148]. Assume S and C are selected feature subset and class labels respectively. This algorithm is based on mutual information, and the purpose of feature selection is to find a feature set S with m featuresf_i, which jointly have the largest dependency on the target class C. It is called Max-Dependency, shown as formula 2.1. $I(.)$ denotes mutual information.

$$\max D(S,C), \quad D = I(\{f_i, i = 1,...,m\}; C) \tag{2.1}$$

As Max-Dependency criterion is not easy to implement since it is often hard to get an accurate estimation for multivariate density which is used to caculate Max-Dependency. An alternative method is to select features based on maximal relevance criterion (Max-Relevance). Max-Relevance is to search features satisfying 2.2.

$$\max D(S,C), \quad D = \frac{1}{|S|}\sum_{f_i \in S} I(f_i; C) \tag{2.2}$$

It is likely that features selected according to Max-Relevance could have rich redundancy. When two features highly depend on each other, the respective class-discriminative power would not change much if either was removed. Therefore, equation 2.3 minimal redundancy (Min-Redundancy) condition can be added to select mutually exclusive features.

$$\min R(S), \quad R = \frac{1}{|S|^2}\sum_{f_i, f_j \in S} I(f_i; f_j) \tag{2.3}$$

The criterion combining the above two constraints is called "minimal-redundancy-maximal-relevance" (mRMR). $\Phi(D,R)$ denotes a operator which combine D and R. It is shown in 2.4.

$$\max \Phi(D,R), \quad \Phi = D - R \tag{2.4}$$

In practice, incremental search methods can be used to find the near-optimal features defined by $\Phi(.)$. Suppose we already have S_{m-1}, the feature set with $m-1$ features. F is original feature set. The task is to select the mth feature from the set $\{F - S_{m-1}\}$. This is done by selecting the feature that maximizes $\Phi(.)$. The respective incremental algorithm optimizes 2.5.

$$\max_{f_j \in F - S_{m-1}} [I(f_j;C) - \frac{1}{m-1} \sum_{f_i \in S_{m-1}} I(f_j;f_i)] \tag{2.5}$$

The algorithm 2.2.2 describes mRMR feature selection scheme. To select the candidate feature set, the algorithm computes the cross validation classification error for a large number of features and finds a relatively stable range of small error. This range is called Ω. The optimal number of features which is denoted as n^* of the candidate set is determined within Ω.

Algorithm 2.2.2: Maximum relevance minimum redundancy algorithm

Input: A training dataset $T = D(F,C)$, number of features to be selected n^*
Output: Selected features S_{n^*}

1 Initialize relative parameters: F← 'initial set of all features', C ← 'class labels', $S = \emptyset$;
2 **for** *each feature* $f \in F$ **do**
3 Calculate $\max_{f_j \in F - S_{m-1}} [I(f_j;C) - \frac{1}{m-1} \sum_{f_i \in S_{m-1}} I(f_j;f_i)]$;
4 Get sequential feature sets $S_1 \subset S_2 \subset ... \subset S_{n-1} \subset S_n$;
5 **for** *each* $k \in S_1, ..., S_k, ...S_n$ **do**
6 Find Ω within which the respective error e_k is consistently small ;
7 Within Ω, find the smallest classification error $e^* = \min e_k$;
8 n^* is chosen as the smallest k that corresponds to e^* ;
9 Return Selected features: S_{n^*}.

2.2.2.3 IG

Information gain (IG) uses Shannon's entropy to measure feature set quality [113]. The information for D given class c_i at the root amounts to

$$I(D) = -\sum_{i=1}^{d} P_D(c_i) \log_2 P_D(c_i) \tag{2.6}$$

the information for D_j due to partitioning D at f is

$$I(D_j^f) = -\sum_{i=1}^{d} P_{D_j^f}(c_i) \log_2 P_{D_j^f}(c_i) \tag{2.7}$$

and the information gain due to feature f is defined as

$$IG(f) = I(D) - \sum_{j=1}^{p} \frac{|D_j|}{|D|} I(D_j^f) \tag{2.8}$$

where $|D|$ is the number of instances in D, and $P_D(c_i)$ are priors for data D.

A feature ordering algorithm using information gain is shown in 2.2.3.

Algorithm 2.2.3: Feature selection according to information gains

Input: A training dataset $T = D(F, C)$, number of features to be selected L
Output: Selected features S

```
Initialize relative parameters: F← f_i, i = 1,2,...n, C ← 'class labels', S = ∅ ;
for each feature f_i ∈ F do
    Calculate its information gain IG(f_i) ;
    insert f_i into S in descending order with regard to IG(f_i) ;
Retain first L feature in S, and delete the others ;
Return Selected features: S.
```

2.2.2.4 ReliefF

Relief is a series of algorithms. It includes Relief, ReliefF and RReliefF. Relief was proposed by Kira and it is a feature weighting algorithm. Relief algorithm is simple and has high efficiency, but it is limited to dealing with two label classification. In 1994, Kononenko expanded it to ReliefF algorithm. ReliefF could process multi-label[88]. In 1997, Kononenko improved ReliefF to RReliefF which could handle continuous value of target attributes[89]. We will introduce ReliefF which is used in chapter 7 to compare with proposed algorithms. In fact, ReliefF's estimate $W[A]$ of attributes A is an approximation of the following difference of probabilities:

$$\begin{aligned} W(A) = W(A) &- \sum_{j=1}^{k} diff(A, R, H_j)/(mk) \\ &+ \sum_{C \notin class(R)} [\frac{p(C)}{1 - p(Class(R))} \sum_{j=1}^{k} diff(A, R, H_j(C))]/(mk) \end{aligned} \tag{2.9}$$

The key idea of ReliefF is to estimate attributes according to how well their values distinguish among instances that are near each other. For that purpose ReliefF for a given instance searches for its two nearest neighbours: one from the same class (called nearest hit) and the other from different class(called nearest miss).

In equation 2.9, $diff(A,R_1,R_2)$ represents the difference of sample R_1 and R_2 on feature A. And it could be calculated by 2.10. $H_j(C)$ is jth near hits in class C.

$$diff(A,R_1,R_2) = \begin{cases} \frac{|R_1[A]-R_2[A]|}{max(A)-min(A)}, if\ A\ is\ continuous \\ 0, if A\ is\ discrete\ and\ R_1[A] = R_2[A] \\ 1, if A\ is\ discrete\ and\ R_1[A] \neq R_2[A] \end{cases} \tag{2.10}$$

ReliefF algorithm is shown in 2.2.4. ReliefF has high efficiency and has no limits of data types. But this algorithm could not remove redundant features.

Algorithm 2.2.4: ReliefF feature selection algorithm

Input: A training dataset D, m ← 'sample size', δ ← 'feature weight threshold', k ← 'Number of nearest hit'

Output: Selected feature subset S whose feature weight larger than δ

Initialize feature weight $W(A) = 0$, $S = \emptyset$;
for *i=1 to m* **do**
    Pick at random an instance $R \in D$;
    Pick at random k nearest hit of R, i.e. H_j (j=1,2,...,k), and k nearest miss of R, i.e. $M_j(C)$;
    for *A=1 to N* **do**
        $W(A) = W(A) - \sum_{j=1}^{k} diff(A,R,H_j)/(mk)$
        $+ \sum_{C \notin class(R)} [\frac{p(C)}{1-p(Class(R))} \sum_{j=1}^{k} diff(A,R,H_j(C))]/(mk)$
for *A=1 to N* **do**
    if $W(A) \leq \delta$ **then**
        $S = S \bigcup A$;
Return Selected features: S.

Some of my proposed algoithms are based on mutual information. Therefore, we choose some algorithms which are based on information entropy to compare, such as DMIFS, mRMR, and IG. And we also investigate the relationship between features and class labels or features in my work. Thus, ReliefF is chosen to compare with my work.

2.3 Network Anomaly Detection by Machine Learning

Anomaly intrusion detection can be seen as a classification problem, and machine learning theory could be used in this field[1, 194]. Figure 2.8 shows the process of knowledge discovery proposed by Richard[73]. From figure 2.8 we can see that feature selection is a data preprocessing task before using machine learning training algorithm. Machine learning algorithms help to create model to distinguish normal and attack instances[22]. There are two types of classification for intrusion detection according to class labels, two class or multiple class [18, 165]. A two class problem regards all attacks as anomaly instances. While a multiple class problem handles all attacks as different labels. Intrusion detection datasets usually have large number of instances and features. And some features may be irrelevant and redundant. Therefore, feature selection could apply on this kind of classification domain [36, 141].

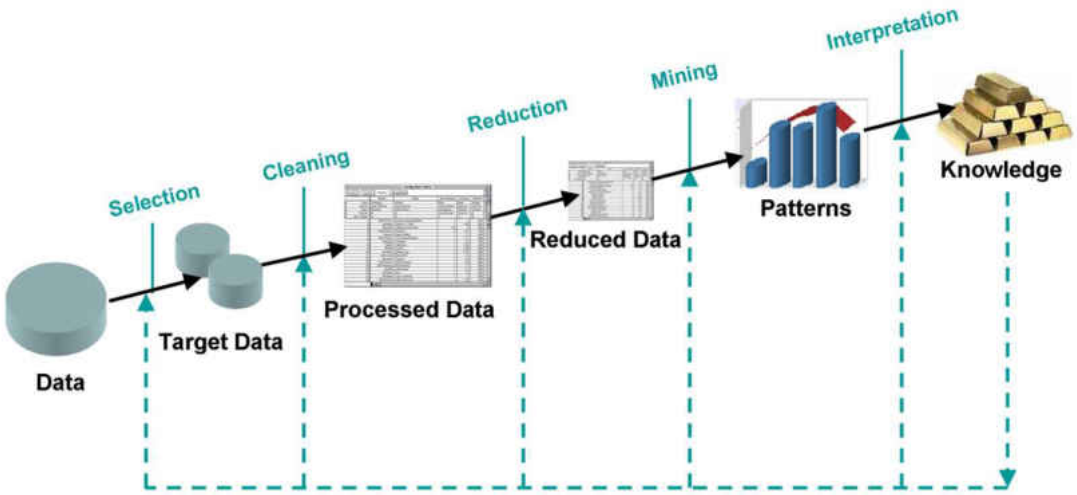

Figure 2.8: Process of knowledge discovery

Feature selection problem can be characterised in the context of machine learning. Assume that $T = D(F, C)$ is a training dataset with m instances and n features, where $D = o_1, o_2, \ldots, o_m$ and $F = f_1, f_2, \ldots, f_n$ are the sets of instances and features. $C = c_1, c_2, \ldots, c_k$ refers to the set of class labels. For each instance $o_j \in D$, it can be denoted as a value vector of features, i.e., $o_j = (v_{j1}, v_{j2}, \ldots, v_{jn})$, v_{ji} is the value of o_j corresponding to the feature f_i.

Given a training dataset $T = D(F, C)$, the task of learning algorithms for classification is to induce a hypothesis h: $F_i \rightarrow C$ from T, where F_i is the value domain of $f_i \in F$. Since there is a limited number of instances in D, there is a classification error $\varepsilon_{F(h)} = |(o, c) \in F | h(o) \neq c|/m$ for each classifier, where $h(o)$ is the predicted class label of o by the hypothesis h. Feature selection can change F, and result in the changing of $\varepsilon_{F(h)}$. After feature selection process, datasets are reduced and then learning methods are used on datasets for classification.

As stated above, there are two learning methods. One is supervised methods and they are based on classifiers, such as C4.5 [133], Bayesian [2, 127], ID3, JRip, PART, SMO and IBK algorithms.

Another one is unsupervised methods and they are based on clustering methods, such as Fuzzy C Means, Sub-Space Clustering (SSC) [144], Density-based Clustering [48], and Evidence Accumulation Clustering (EAC) techniques [44]. One advantage of unsupervised methods is that they could detect unknown attacks[104, 150].

As mentioned above, feature selection algorithms could be classified into three categories, filter, wrapper and hybrid. Filter methods are not depend on learning algorithms and they have better performance on dealing with massive data or online data. And filter methods evaluate whether features are important or not by their inherent characteristics, such as distance function, rough set, mutual information, independent component analysis and statistical correlation coefficient [173, 29]. Relief algorithm proposed by Kira and improved algorithm ReliefF are based on Euclidean distance [87]. Hu introduced Neighborhood Margin(NM) and Neighborhood Soft Margin(NSM) to measure minimum distance. And he uses NSM to evaluate the quality of candidate subsets [64]. Zhang proposed a consistency measurement which is named Pairwise Constraints to evaluate subsets [191]. Rough set is proposed by Pawlak in 1982 and it is an effective way to deal with the uncertain information[106]. Richard presented new feature selection algorithms combined rough and fuzzy sets[73]. Abe and Kudo put forward a method to select the most relevant features with class labels by Bayes boundary error [66].

Wrapper methods use machine learning algorithm as the prediction performance evaluation criteria of the feature subset. Huang proposed a hybrid genetic algorithm based on information theory[67]. And he used the algorithm and classifier to get subsets. Jarvis and Goodacre use genetic algorithm as well to select the best subset[41].

Kabir put forward a algorithm based on neural network[79]. Inza takes Bayes network into wrapper algorithms and get improved classification performance[69].

The degree of relevance within a feature subset is very important to the performance of feature selection. Generally, there are two methods to measure relevance between features, linear correlation measurement and correlation measurement based on information theory. BIF feature selection algorithm using mutual information to measure the degree of relevance between the features and the class labels, and output K highest degree features as the optimal feature subset [72]. This method can effectively eliminate the irrelevant features, but the selected features still have a large number of redundant features. Battiti put forward a feature selection algorithm based on mutual information (MIFS) [13]. This algorithm uses mutual information to measure the relevance between features and class labels. At the same time, it also calculates relevance degree between a candidate feature and the selected feature set. However, the MIFS algorithm has lower robustness and when facing the redundant features is highly correlated to class labels. Kwak and Choi introduced MIFS-U algorithm which uses uncertainty coefficient to represent relevance degree of features [93]. Peng proposed Max-Relevance and Min-Redundancy algorithm and it uses mutual information to evaluate features [148]. Lee introduced information gain and divergence-based feature selection algorithm. The algorithm obtains feature subset by deleting the redundant features while maintaining the information gain [99]. [3] proposed mutual information-based feature selection method results in detecting intrusions with higher accuracy.

Unsupervised methods also could be used to perform feature selection. For example, mRR uses conditional mutual information to measure the relevance between features. And correlated instances are clustered based on hierarchical clustering technology, so as to remove the redundant features [123]. ACA uses a function in information theory to measure dependancy between features. Features dependent on each other will form a cluster and then it selects representive features from clusters. Thus, the redundant features could be removed [9]. FCBF algorithm measures the relevance of features by symmetrical uncertainty. And it also proposed a feature selection algorithm which can remove redundant features effectively [190].

Chapter 3, 4, 5 propose four supervised algorithms and chapter 6 describes two unsupervised algorithms. And they will be introduced specifically in the following chapters. All of the supervised algorithms use filter method and some of them

take advantage of mutual information to measure the relevance between features. Two supervised algorithms improve Battiti's MIFS algorithm and mRMR algorithm respectively. Another two supervised algorithms take advantage of feature grouping and hierachical clustering and combine them together. One unsupervised algorithm combine supervised algorithm with itself. Another clustering algorithm improves evidence accumulation clustering algorithm. All the six algorithms are tested on KDD 99 dataset and intrusion detection classification is considered as a two class problem. In other words, every instance in KDD 99 dataset is labeled either anomaly or normal data.

2.4 Summary

In this chapter, intrusion detection and feature selection are introduced. Intrusion detection is not a new research area but it is an important area since new challenges and new attacks emerge continuously. The history and challenges are briefly described in this chapter, and some solutions for IDS are also introduced.

Since both instance and features in IDS data usually very huge, we use feature selection method to reduce dimensionality and remove irrelevant and redundant features. And feature selection could also help improving classification accuracy. In this chapter, we introduced feature selection models and some feature selection algorithms. These algorithms will be used to compare with proposed algorithms in following chapters. Machine learning methods using on network anomaly detection are briefly set forth in this chapter as well. Some literatures mentioned in this part are used to classify or cluster for IDS.

Chapter 3

Modified Mutual Information-based Feature Selection for Intrusion Detection Systems in Decision Tree Learning

THIS chapter proposes a modified mutual information-based feature selection algorithm (MMIFS) for intrusion detection on the KDD Cup 99 dataset. The C4.5 classification method was used with this feature selection method. In comparison with dynamic mutual information feature selection algorithm (DMIFS), we can see that most performance aspects are improved. Furthermore, this chapter shows the relationship between performance, efficiency and the number of features selected.

3.1 Mutual Information-based Feature Selection Method Introduction

3.1.1 Mutual Information

When Information theory was firstly designed to measure the size of the information magnitude in data communication, entropy is an important measurement [188, 46]. It is used to quantify the uncertainty of random variables, and also to effectively scale

the information amount among them. We only talk about finite random variables with discrete values within this chapter [45].

If X is a random variable with discrete values, its entropy is defined as

$$H(X) = -\sum_{x \in X} p(x) log p(x) \tag{3.1}$$

where $H()$ is entropy, and $p(x) = P_r(X = x)$ is the probability density function of X. Note Entropy is dependent on the distribution of the probability of the random variable.

If conditional entropy is defined as the uncertainty reduction in one variable while the other is known, assume variable Y is given, the conditional entropy $H(X|Y)$ of X with respect to Y is

$$H(X|Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) log p(x|y) \tag{3.2}$$

where $p(x,y)$ is the joint probability density function and $p(x|y)$ is the posterior probabilities of X given Y.

Also the joint entropy $H(X,Y)$ of X and Y is

$$\begin{aligned} H(X,Y) &= H(X) + H(Y|X) \\ &= H(Y) + H(X|Y) \\ &= \sum_{y \in Y} \sum_{x \in X} p(x,y) log p(x,y) \end{aligned} \tag{3.3}$$

In order to quantify the amount of information shared by two variables X and Y, a termed mutual information $I(X;Y)$ is introduced as

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= \sum_{y \in Y} \sum_{x \in X} p(x,y) log \frac{p(x,y)}{p(x)p(y)} \end{aligned} \tag{3.4}$$

When X and Y are unrelated, the value of $I(X;Y)$ is 0. While $I(X;Y)$ is high, it means X and Y are closely related. The mutual information is applicable in the evaluation of any arbitrary dependency between random variables. Within this chapter, we only compute the mutual information between two variables, and scale the mutual dependence between them [163].

3.1.2 Application to Feature Selection

The central assumption when using a feature selection technique is that the data contains many redundant or irrelevant features. Redundant features are those which provide no more information than the currently selected features, and irrelevant features provide no useful information in any context. Figure 3.1 shows analysis of features relevance and redundancy[182]. Feature selection is used to remove irrelevant and redundant features from datasets. Irrelevant features can be easily removed and strongly relevant features are easily found by relevance analysis. So removing redundant features is a very important task for optimal feature selection process. And redundant features are usually in weakly relevant features. Our work is trying to remove irrelevant and redundant features effectively. In KDD99 dataset, some features may be irrelevant and others may be redundant since the information they add is contained in other features. These extra features can increase computation time for creating classifications, and can have an impact on the accuracy of the classifier built. For this reason, this classification domain seems to be suitable for the application of feature selection methods. These methods are centered in obtaining a subset of features that adequately describe the problem at hand without degrading performance.

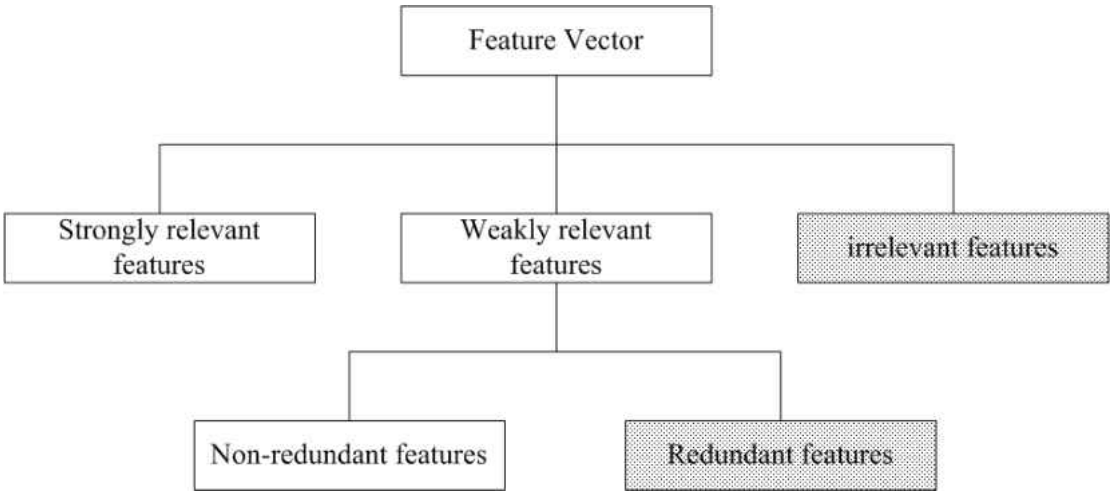

Figure 3.1: Features relevance and redundancy analysis

To verify that there are irrelevant and redundant features in KDD Cup 99 dataset, Correlation based Feature Selection (CFS) is used to select 8 features by Weka. Two performance measures (precision and F-measure) were calculated which will specifically be discussed in section 3.3.2 and we used four classification methods to

calculate the two performances. Figure 3.2 shows the precision comparison between 41 features and 8 features by normal and anomaly types respectively. Similarity, figure 3.3 describes the other performance F-measure.

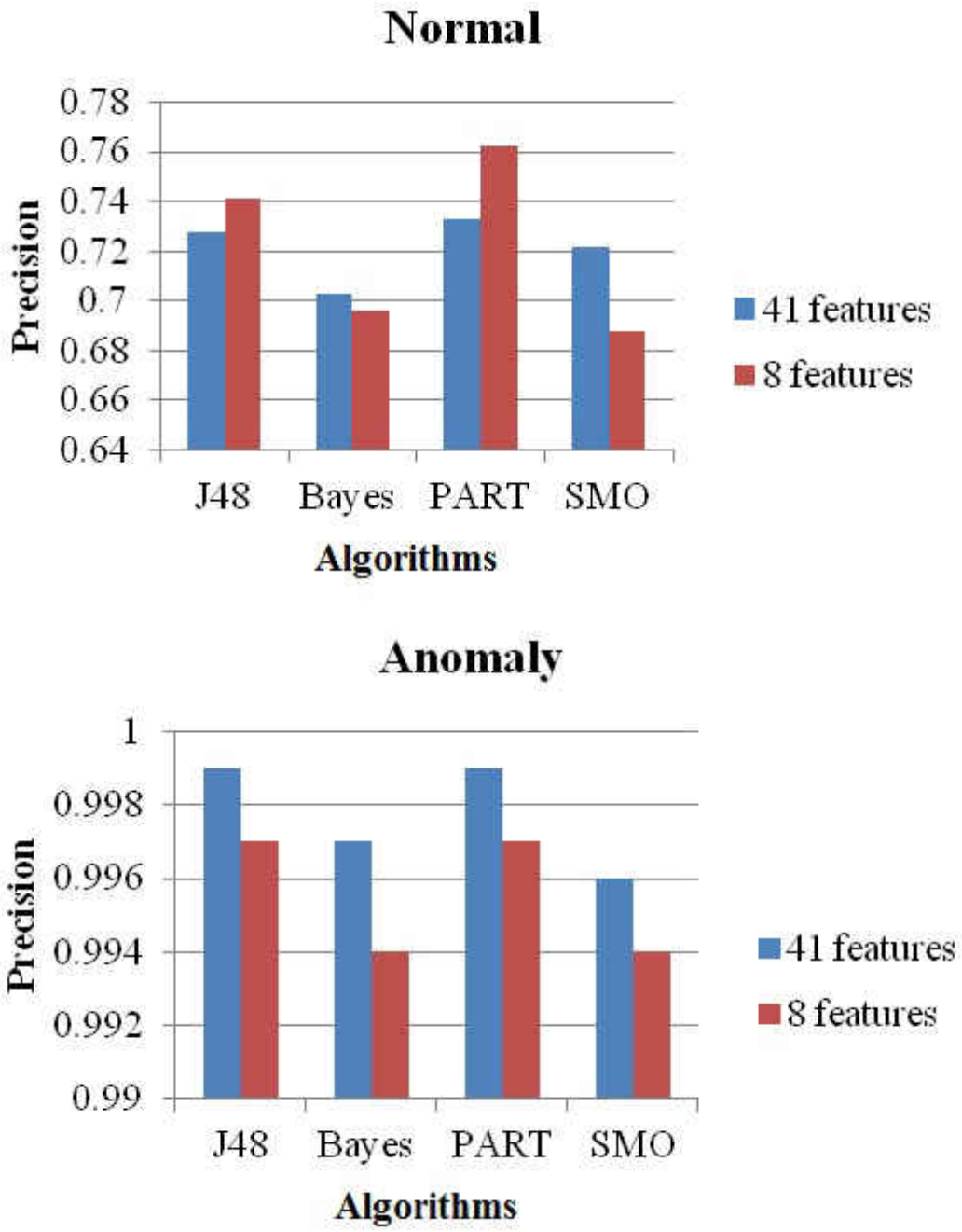

Figure 3.2: Precision comparison chart between all features and selected features

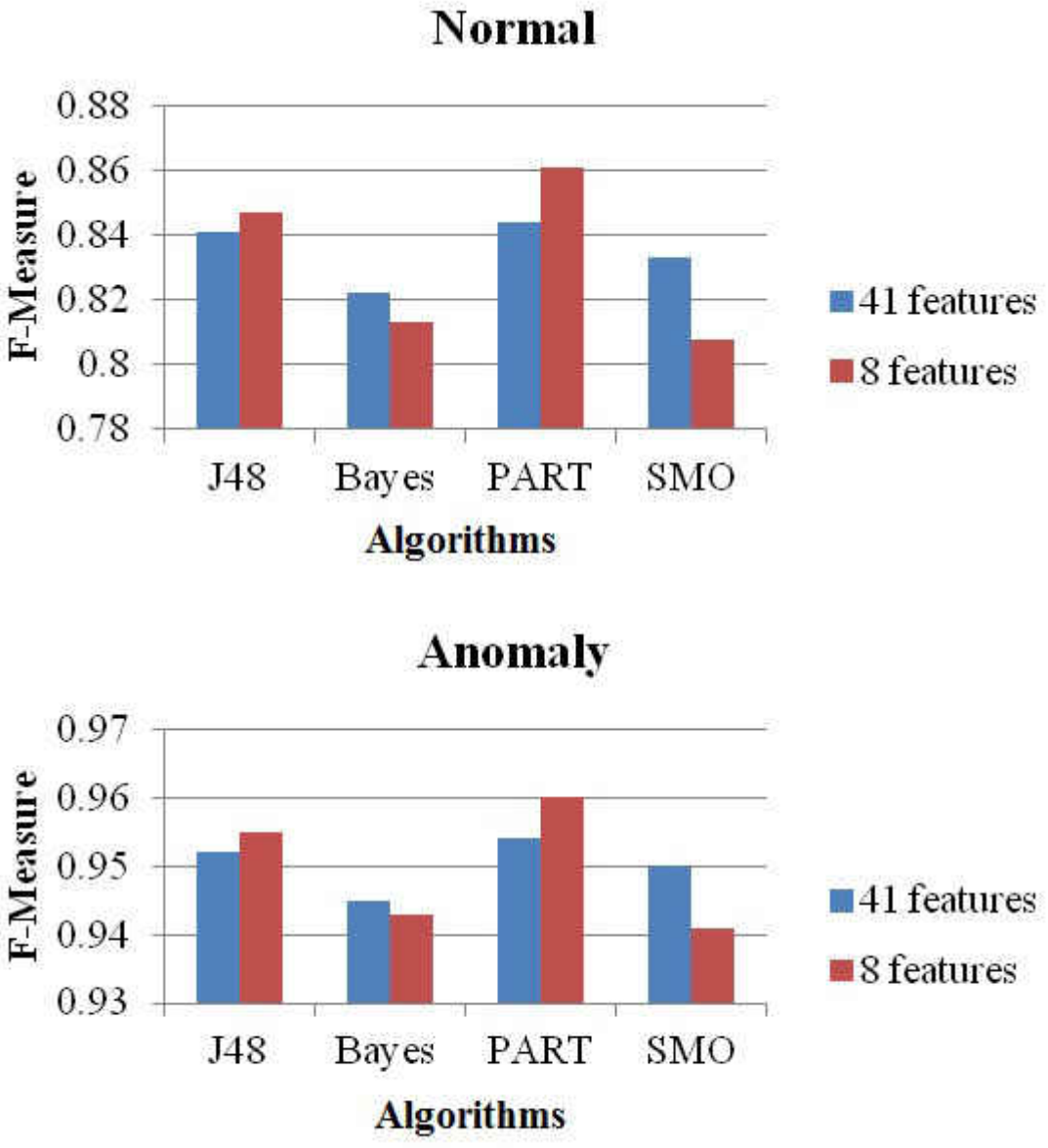

Figure 3.3: F-measure comparison chart between all features and selected features

The two figures show for each classification method, that the performance comparison between 8 and 41 features is quite close. For J48 and PART methods, the performance with 8 features is actually improved. Another advantage of selecting features is the running time is shorter than using all features. We will show the computation time comparison in section 3.3.2.

3.2 Modified Mutual Information-based Feature Selection for Intrusion Detection Systems

First of all, the mutual information between each feature and class label in the KDD99 dataset is calculated. The results are shown in figure 3.4. Figure 3.4 shows that feature 5 has the largest mutual information value. This means that feature 5 and the class label have the largest correlation.

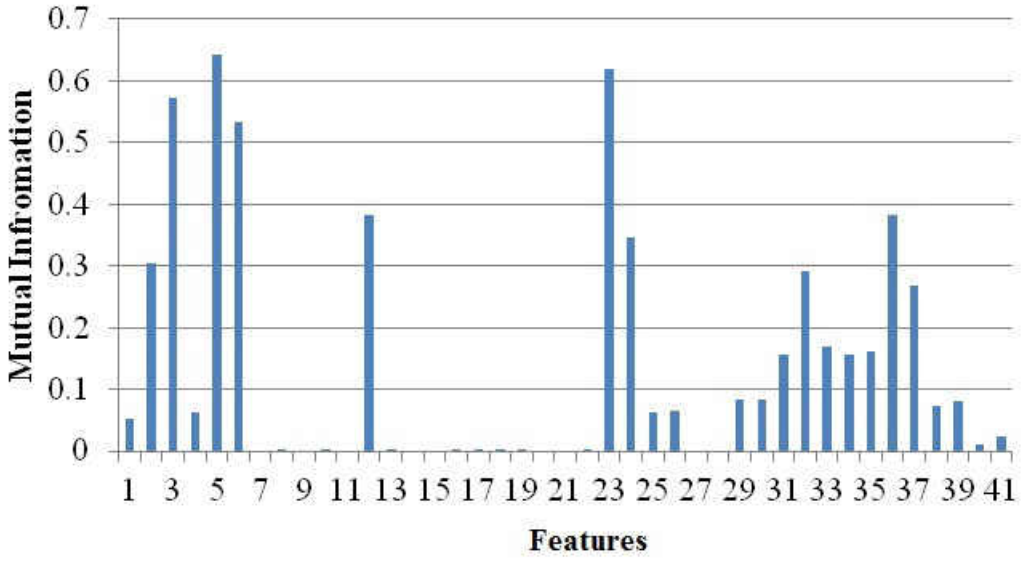

Figure 3.4: Mutual information of between each feature and class label in KDD99 dataset

We could rank the features by mutual information from figure 3.4. But we could not select the features according to this way. Take features 5, 12 and 23 as an example, let C represent class label and mutual information between the three features and C are $I(f_5;C) = 0.6424$, $I(f_{12};C) = 0.381$, $I(f_{23};C) = 0.6179$. In descending order, the three are sorted as f_5, f_{23}, f_{12}. But after f_5 is selected, we

should delete the correct instances induced by f_5. Battiti proposed an evaluation function considering the mutual information between features, which is shown by formula 3.5, along with the method called mutual information-based feature selection (MIFS). In this case, the mutual information between f_5 and f_{23} is $I(f_5, f_{23}) = 1.472$ and the mutual information between f_5 and f_{12} is $I(f_5, f_{12}) = 0.5436$. According to Battiti's evaluation function, f_{12} will be selected, rather than f_{23}. In 2009, Huawen Liu proposed a dynamic mutual information method called DMIFS. And DMIFS improved MIFS in respect to some performance.

$$I(f_i; C) - \beta \sum_{f_s \in S} I(f_i; f_s) \tag{3.5}$$

In formula 3.5, f_i represents each feature in a set and fs denotes a selected feature in a selected feature set S. There is a parameter β and Battiti suggested it should be between 0.5 and 1. But in our study, we think the parameter should be related to mutual information between each feature and class label, rather than a fixed value. So we put forward an improved algorithm named MMIFS as follows.

Algorithm 3.2.1: Modified Mutual Information based Feature Selection algorithm

Input: A training dataset $T = D(F, C)$
Output: Selected features S

1 Initialize relative parameters: F← 'initial set of all features', C ← 'class labels', $S = \emptyset$;
2 For each feature $f_i \in F$, compute the mutual information of the features with the class labels $I(f_i; C)$;
3 Selection of the first feature: find the f_i that maximizes the $I(f_i, C)$, then $S = S \cup f_i$, $F = F \setminus f_i$;
4 **while** *Desired number of selected features is not achieved* **do**
5 Computation of the mutual information between features: for all pair of features (f_i, f_s), where $f_i \in F$ and $f_s \in S$, compute $I(f_i; f_s)$;
6 Selection of the next feature: choose the feature fi as the one that maximizes $I(f_i; C) - \sum_{f_s \in S} I(f_i; C) * I(f_i; f_s)$;
7 return Selected features: S.

3.3 Experimental Results

3.3.1 Implemented System

C4.5 is used to classify the feature set that was selected by applying MMIFS. C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan and it is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. C4.5 uses the concept of information gain to make a tree of classificatory decisions with respect to a previously chosen target classification. The information gain can be described as the effective decrease in entropy resulting from making a choice as to which attribute to use and at what level.

The classification is based on six measures: True Positive Rate (TPR), False Positive Rate (FPR), Precision, Total Accuracy, Recall, F-Measure. The six measures are calculated by True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), as follows.

True positive rate (TPR): TP/(TP+FN), also known as detection rate (DR) or sensitivity or recall.

False positive rate (FPR): FP/(TN+FP) also known as the false alarm rate.

Precision (P): TP/(TP+FP) is defined as the proportion of the true positives against all the positive results.

Total Accuracy (TA): (TP+TN)/(TP+TN+FP+FN) is the proportion of true results (both true positives and true negatives) in the population.

Recall (R): TP/(TP+FN) is defined as percentage of positive labeled instances that were predicted as positive.

F-measure: 2PR/(P+R) is the harmonic mean of precision and recall.

In our experiments, we need to determine the desired feature numbers which we expect to select in KDD Cup 99 dataset. Thus, we calculated total accuracy of different feature numbers which are obtained by MMIFS. The results are shown in figure 3.5.

We can see from the figure that we tested 13 features obtained by MMIFS. The reason we tested 13 features is that less or equal than 13 selected features of KDD

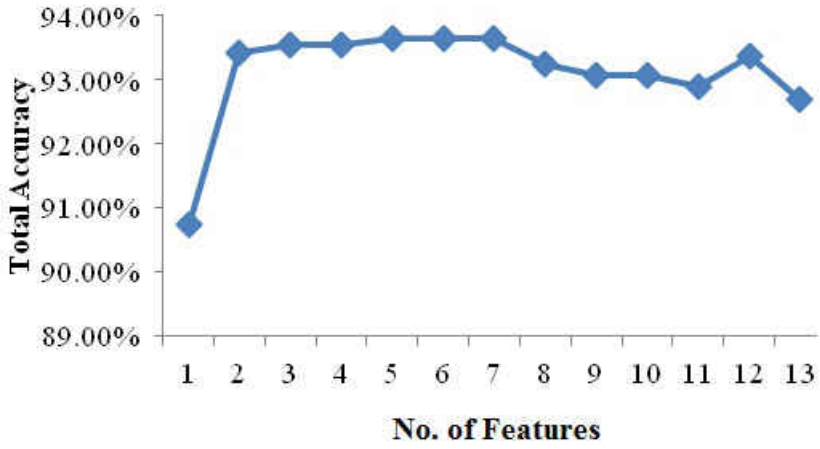

Figure 3.5: Total accuracy of different feature numbers

99 dataset could achieve best performance and relatively less computation time according to the feature selection algorithms we used before(e.g.CFS). The total accuracy does not increase as the numbers rise. The reason is because there are many noisy and redundant features in the dataset.

From figure 3.5, we can see that the total accuracy is between 93% and 94% if we select 2 to 13 features. And the accuracy is very close for different feature numbers. But considering the large number of instances in the KDD 99 dataset, a small improvement in accuracy will result in many instances being correctly classified.

A range of features between 2 and 13 could be used for comparison. But when we used DMIFS to get the features, we realised if the desired numbers are small, most of the features are the same as we got by MMIFS. Thus, we choose 10 features to compare the algorithms since the algorithm (DMIFS) which is compared with MMIFS could achieve the best performance when it select 10 features.

3.3.2 Results

As we disscussed in section 2.3, intrusion detection can be considered as a two class problem or a multiple class problem. In this experiment, we regard all attack types as anomaly patterns and the other class is a normal pattern, addressing intrusion detection as a two class problem.

In the following subsection, C4.5 is used to classify the dataset and compare the performance between DMIFS and MMIFS. C4.5 is better than some other classification

algorithms and comparison between C4.5 and 3 other algorithms by using 10 selected features shows in table 3.1. From table 3.1, we can see that C4.5 is much better than other 3 algorithms. Though some other methods could achieve the same or even better performance than C4.5, such as SVM or neural network. But they take longer computation time. The experiments were conducted on the KDD 99 dataset and performed on a Windows machine having configuration and Intel (R) Core (TM) i5-2400 CPU@ 3.10GHz, 3.10 GHz, 4GB of RAM, the operating system is Microsoft Windows 7 Professional. We have used an open source machine learning framework Weka 3.5.0. We have used this tool for performance comparison of our algorithm with other classification algorithms. Table 3.2 shows the specific comparison and it indicates that most of the performances are improved by MMIFS compared to DMIFS and Battiti's method, such as precision and F-measure. The experiment is executed 10 times and the differences in performance of different methods are statistically evaluated using paired t-test with two-tailed $p = 0.01$. MMIFS could achieve statistically better result. The total accuracies for these three methods are 92.65%, 92.94% and 93.02% respectively.

Suppose there are m instances and n features in training dataset. The time complexity of MMIFS, DMIFS and Battiti's algorithms are $O(mn^2)$, $O(mn^2)$ and $O(mn)$ respectively.

From the comparison results, we can see that MMIFS could have better performance than other two algorithms when they all select 10 features. The first two rows show the performance of C4.5 with all 41 features. And the next four rows describe results of Battiti's method and DMIFS. They have the same results since they select the same 10 features. MMIFS get the best performance although differences are not very large.

Another advantage for applying feature selection methods on KDD 99 dataset is the saving in computation time. In C4.5 algorithm, we need to build a model from the KDD 99 training dataset first and then evaluate the model on the test dataset. Figure 3.6 describes the time taken to build model comparison by the different feature numbers. Feature selection algorithm is used on 10 percent KDD training dataset first. Then we used 10 percent of KDD training dataset to create decision tree model first and re-evaluated on test dataset. And some computation time is spend for it. Figure 3.7 illustrates the total time comparison by different feature numbers.

Table 3.1: Comparison Results between C4.5 and other 3 Algorithms

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
C4.5	0.99	0.084	0.741	0.99	0.848	normal
	0.916	0.01	0.997	0.916	0.955	anomaly
Naive Bayes	0.974	0.107	0.688	0.974	0.806	normal
	0.893	0.026	0.993	0.893	0.94	anomaly
OneR	0.984	0.111	0.681	0.984	0.805	normal
	0.889	0.016	0.996	0.889	0.939	anomaly
LogitBoost	0.976	0.093	0.718	0.976	0.828	normal
	0.907	0.024	0.994	0.907	0.949	anomaly

Table 3.2: Comparison Results between DMIFS and MMIFS Algorithm

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
C4.5	0.994	0.09	0.728	0.994	0.841	Normal
	0.91	0.006	0.999	0.91	0.952	Anomaly
C4.5 with Battiti's	0.993	0.086	0.736	0.993	0.846	normal
	0.914	0.007	0.998	0.914	0.954	anomaly
C4.5 with DMIFS	0.993	0.086	0.736	0.993	0.846	normal
	0.914	0.007	0.998	0.914	0.954	anomaly
C4.5 with MMIFS	0.99	0.084	0.741	0.99	0.848	normal
	0.916	0.01	0.997	0.916	0.955	anomaly

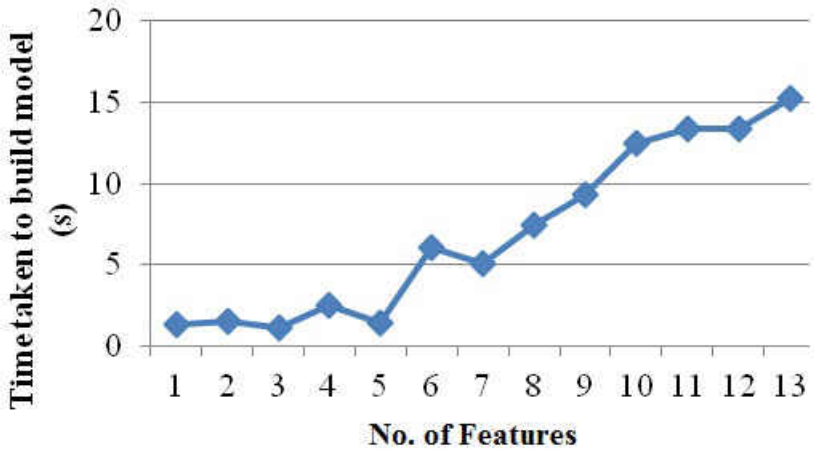

Figure 3.6: Time taken to build model comparison chart

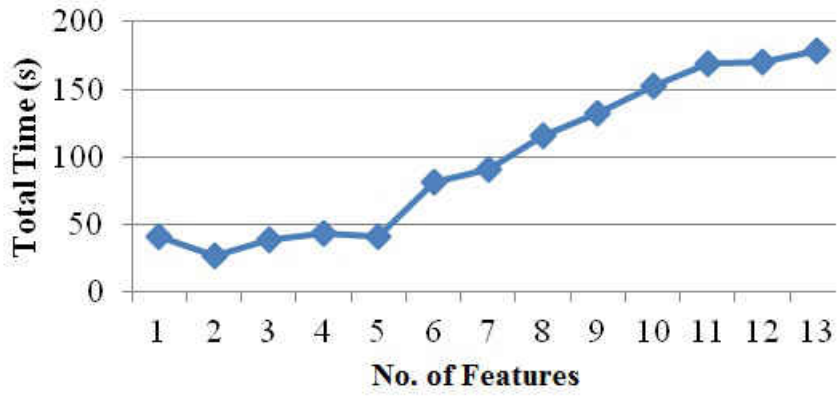

Figure 3.7: Total time comparison chart

We can see from figure 3.6 and figure 3.7 that as the number of features increases, the calculation time increases significantly. It indicates that the computation time is greatly affected by the numbers of features.

3.4 Summary

This chapter proposed a new feature selection method and the main improvement of this work is that it modifies the mutual information feature selection algorithm by changing the weighting parameter. We tested this method on the KDD 99 dataset and compared the results with the DMIFS algorithm. The results show that most of the performance indicators are improved. Future work will evaluate the algorithm against other datasets which have less noise and less redundant features. The value of the weighting parameter may not be optimum, and so further study will attempt to find values of the parameter that produce the best results. Finally, we will try to compare the method based on correlation coefficient of features with the method based on mutual information.

Chapter 4

Feature Grouping for Intrusion Detection based on Mutual Information

FEATURE grouping is using the relationship of features in a dataset to compose groups and design selection strategy to select a feature or features from a group[116]. We could take feature grouping as a kind of feature selection. In particular, feature grouping that allows the selection of multiple features by one go is applicable to the dataset with a high dimensionality.

In this chapter, two algorithms are proposed and they are both using feature grouping method and based on mutual information. One is mutual information-based feature grouping algorithm which is introduced in 4.1. The other one is feature grouping by agglomerative hierarchical clustering based on mutual information, which is presented in 4.2.

4.1 Mutual Information-based Feature Grouping Method

4.1.1 Application to Feature Selection

As stated in 2.3, feature selection could help for classification task. For example, Battiti's work is based on mutual information to select features showed in 3.5.

Formula 3.5 can be used to select the next feature. β is a parameter and determined empirically and Battiti has proposed a value between 0.5 and 1 for β. This algorithm indicates that feature selection should consider not only the mutual information between each feature and class label but also the mutual information between each feature and selected features.

If there are n features in the dataset and f_i is the feature i, then $M_i(f_i;F)$ denotes the mutual information between f_i and all the other features. And it shows in formula 4.1.

$$M_i(f_i;F) = \sum_{j=1 j\neq i}^{n} I(f_i;f_j) \tag{4.1}$$

When i=1,2,3,...,n, $SUM_{MI} = [M_i(f_i;F)]$ denotes the vector set of C.

Feature grouping could been seen as a kind of feature selection. We could measure the relationship between one feature and other features by some methods. And then we can use it to compose groups. Features who have similar metrics be put into will in one group.

4.1.2 Selecting Strategy of Feature Grouping

Feature Grouping is highly beneficial in learning with high dimensional data. It reduces the variance in the estimation and improves the stability of feature selection. Furthermore, it could help in data understanding and interpretation as well. The purpose of feature grouping is creating groups for candidate selecting features and selecting one or more features from certain groups to represent the group.

Clustering methods could be used to create groups since they select data in one cluster by specific metrics. Different clustering methods and metrics could compose different cluster constructions. Number of clusters affects how many features will be selected. For example, different strategies could be adopted if we expect to select 8 features from a dataset. We could create 8 groups by a clustering method and select 1 feature in each group. Or we could construct 4 groups and select 2 features per group instead. Figure 4.1 is shown one example of feature grouping strategy. Moreover, we could select different numbers of features in different groups. For example, hierarchical clustering method could be used to create groups in this work, we chose the selecting 1 feature from each group strategy. This strategy is simple

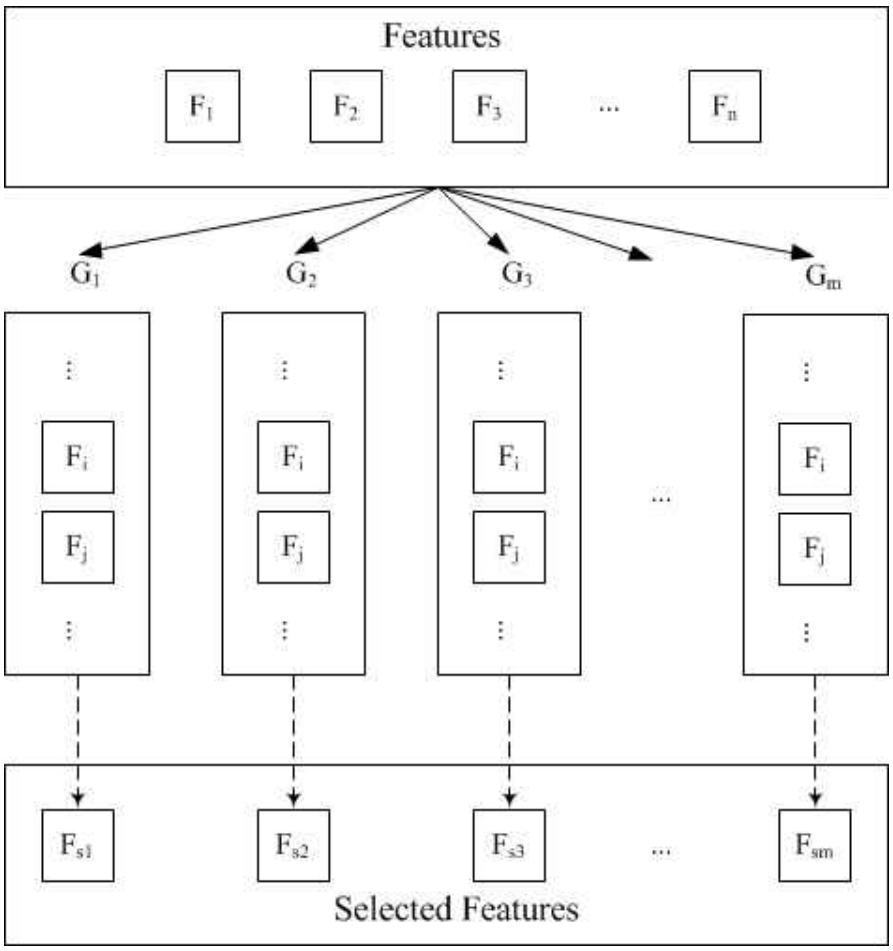

Figure 4.1: Selecting strategy example of feature grouping

and easy to implement. And another reason is there might be only one feature in one group by using agglomerative hierarchical clustering method.

From figure 4.1 we can see there are n features $F_1, F_2, ..., F_n$. And they compose m groups $G_1, G_2, ..., G_m$ by using a specific method. Then in each group, we select one feature and get selected features set $F_{s1}, F_{s2}, ..., F_{sm}$.

4.1.3 Feature Grouping based on Mutual Information Algorithm

Feature selection can be improved on through Feature Grouping based on Mutual Information (FGMI) as follows.

From the algorithm 4.1.1, it can be seen that the number of features selected by this algorithm depends on the number of groups. The mutual information between

Algorithm 4.1.1: Feature Grouping based on Mutual Information

Input: A training dataset $T = D(F, C)$, G
Output: Selected features S

1 Initialize parameters: F← 'initial set of all features', C ← 'class labels', $S = \varnothing$;
2 For each feature f_i, calculate the mutual information between f_i and all the other features in F, then sum the results together and it can be calculated by formula 4.1, and finally get a vector SUM_{MI};
3 Use Fuzzy C Means algorithm on SUM_{MI} to get G groups ;
4 For each group g in G, calculate mutual information between each feature and class label in C, and then find the maximum value M_g in each group ;
5 Select feature f_s which has the M_g in each group, and put f_s into S, $S \leftarrow' fs'$;
6 Return the set containing the selected features: S.

each pair of features is calculated and Fuzzy C-Means algorithm is used to compose the groups. Fuzzy C-Means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters. This method (developed by Dunn in 1973 and improved by Bezdek in 1981) is frequently used in pattern recognition and unsupervised classification.

At first, the proposed algorithm calculates the mutual information between each feature and all the other features and adds them together, denoted as SUM_{MI}. Then, it ranked the SUM_{MI} by Fuzzy C-Means algorithm to get G groups. Moreover, in each group, the algorithm computes mutual information between each feature and class label and get the maximum one. At last, select the feature which has the maximum value.

From the process of FGMI, we can see that G decides how many features will be selected by this algorithm. FGMI composes G groups and selects one feature from each group. In other words, the algorithm will select G features. FGMI requires users to input G at first as FGMI does not attempt to decide how many features should be selected by itself. Someone might argue this is a disadvantage of the algorithm, but FGMI is very efficient in computation time. The reason is that algorithms that automatically calculate the optimum number of selected features need to add performance evaluation or deduce part in the algorithms. However, FGMI would be able to use performance evaluation of selected features to find the best G. FGMI is appropriate for datasets who have large number of instance such as KDD 99.

To compose groups, we could use many methods, such as divide SUM_{MI} into G groups on average. But after ranked SUM_{MI}, we realised data in SUM_{MI} is

unbalanced and nearly half of the values are quite low. So we chose to use clustering algorithm to compose groups. Fuzzy C-Means is used to compose G groups in FGMI. This algorithm could divide a set $X = \{x_1, x_2, ..., x_n\}$ into C clusters and make objective function get minimum value. Data are bound to each cluster by means of a Membership Function, which represents the fuzzy behaviour of this algorithm. Data in SUM_{MI} are one-dimensional and unbalanced. After using Fuzzy C-Means on SUM_{MI}, most low values in SUM_{MI} will go into one group. In contrast, high values in SUM_{MI} will go into different groups.

4.2 Feature Grouping by Agglomerative Hierarchical Clustering based on Mutual Information

In this section, feature grouping by agglomerative hierarchical clustering based on mutual information(FGMI-AHC) is described in detail. The basic idea is grouping the features by agglomerative hierarchical clustering method, and then selecting features from the groups. As we used a clustering method to construct groups, cluster and group have the same meaning in the following formulation.

4.2.1 Hierarchical Clustering

Hierarchical clustering is a clustering method to build a hierarchy of clusters[61]. There are two types of strategies for hierarchical clustering, agglomerative and divisive[138]. Agglomerative is a bottom-up approach where initially every data item constitutes its own cluster, and pairs of clusters are merged as one moves up the hierarchy[31, 85]. Divisive is a top-down approach and all data is part of the initial cluster and splits are performed recursively as one moves down the hierarchy[142].

In order to decide which clusters should be combined or split, a measure of dissimilarity between sets of observations is required. In most methods of hierarchical clustering, this is achieved by use of an appropriate metric and a linkage criterion. In this chapter, an agglomerative hierarchical clustering algorithm is used based on linkage rule. The rule means different distance metrics and methods which could be used in linkage. Take the test on KDD 99 for example, figure 4.2 shows dendrogram of agglomerative hierarchical clustering on KDD99 by median distance. Different metric will results in different dendrogram. Figure 4.3 describes dendrogram of agglomerative hierarchical clustering on KDD99 by inner squared distance.

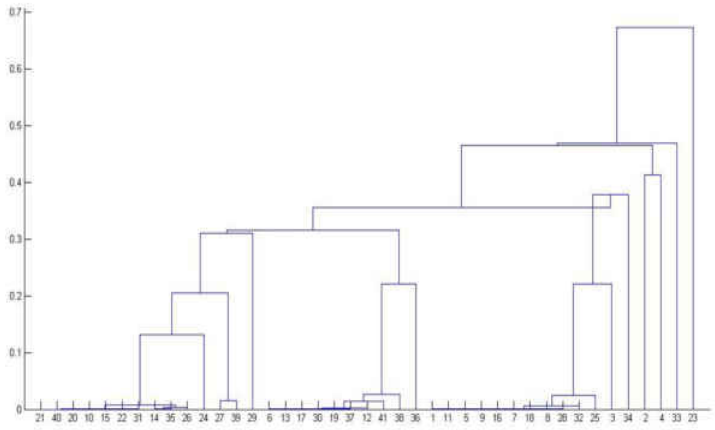

Figure 4.2: Dendrogram of agglomerative hierarchical clustering on KDD99 by median distance

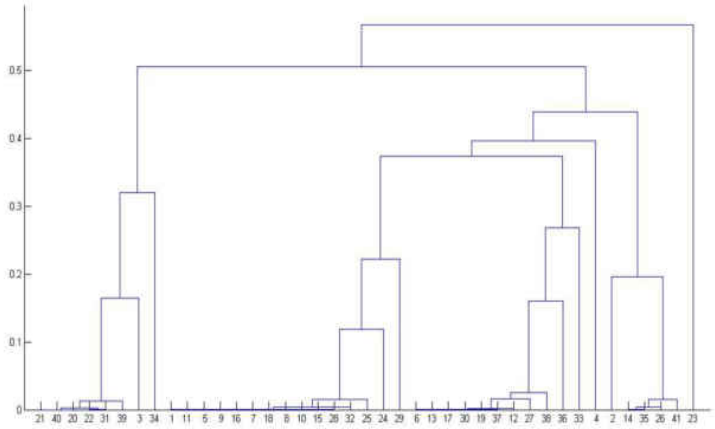

Figure 4.3: Dendrogram of agglomerative hierarchical clustering on KDD99 by inner squared distance

There are some methods for computing distance between clusters, such as average, centroid, complete, median, single, ward, weighted and so on. Different method will result in different cluster tree, such as figure 4.2 and figure 4.3.

4.2.2 Implemented Algorithm

In this section, we will show the algorithm put forward by this chapter. The detailed algorithm is shown as follows.

Algorithm 4.2.1: Feature Grouping based on Agglomerative Hierarchical Clustering Algorithm

Input: A training dataset $T = D(F, C)$, number of clusters n.
Output: Selected features S

1 Initialize parameters: F← 'initial set of all features', C ← 'class labels', $S = \varnothing$;
2 Calculate the mutual information of every pair of features f_i and f_j in F, denote as $I(f_i; f_j)$;
3 Create hierarchical cluster tree by using agglomerative hierarchical clustering method base on $I(f_i; f_j)$;
4 Construct clusters from a hierarchical cluster tree by given n ;
5 For each cluster, calculate mutual information between each feature and class label in C, and then find the maximum value M_c ;
6 Select feature f_s which has the M_c in each group, and put f_s into S, $S \leftarrow' f_s'$;
7 Return selected features: S.

First of all, the algorithm decides initialization parameters and F is a set of all the features in the training dataset. And C denotes class labels and n represents clusters number. Then, the algorithm calculates the mutual information of every pair of features in F and composes a matrix based on them. After that, it creates a hierarchical cluster tree based on the matrix by using an agglomerative hierarchical clustering method. Moreover, it constructs clusters from a hierarchical cluster tree by given n. And n clusters mean n groups containing candidate features. Furthermore, in each cluster, it calculates mutual information between each feature and class label in C, and then finds the maximum value M_c. Finally, it selects feature f_s which has the M_c in each group, and put f_s into S.

There are some distances could be used to compute between pairs of objects data when we using AHC. Such as Euclidean, chebychev,cosine, correlation, hamming and so on. In this algorithm, we take mutual information between each pair of features as

their distance. Thus some of methods for computing distance between clusters could not be used since they are appropriate for Euclidean distances only. That means centroid, median and and ward could not be used in this algorithm. And complete and single mean furthest and shortest distance respectively. In this algorithm, we use average menthod which means unweighted average distance. The reason is average method is least affected by abnormal data.

As stated in 4.1.3, this algorithm also requires users to input number of clusters n first. And we select one feature from each cluster. It means the number of clusters is equal to the number of features you will select. This algorithm constructs a maximum of n clusters and finds the smallest height at which horizontal cut through the tree leaves n or fewer clusters.

4.3 Experimental Evaluation - Comparison with Other Approaches

4.3.1 Results by FGMI

As described in the section 4.1.3, the number of features obtained from the algorithm depends on the number of groups. Fuzzy C Means algorithm is used to divide the ranked vector SUM_{MI}. From our previous work in this area, we tested some feature selection algorithm on KDD 99 dataset,such as CFS. If features are selected between 8 and 14, it could achieve better performance on KDD 99 dataset, and the performance evaluations are as follows.

C4.5 algorithm is used to classify the dataset. The classification performances are usually denoted by six measures which are presented in 3.3.1 specifically. In this section, we will use these measures to evaluate the algorithms' performance. Figure 4.4 shows the TPR comparison by different number of features. Figure 4.5 describes the FPR comparison by different number of features. Figure 4.6 illustrates the precision comparison by different number of features. Total Accuracy and F-Measure comparison chart by different number of selected features are shown in figure 4.7 and figure 4.8 respectively.

The experiment is executed 10 times and the differences in performance of different methods are statistically evaluated using paired t-test with two-tailed p

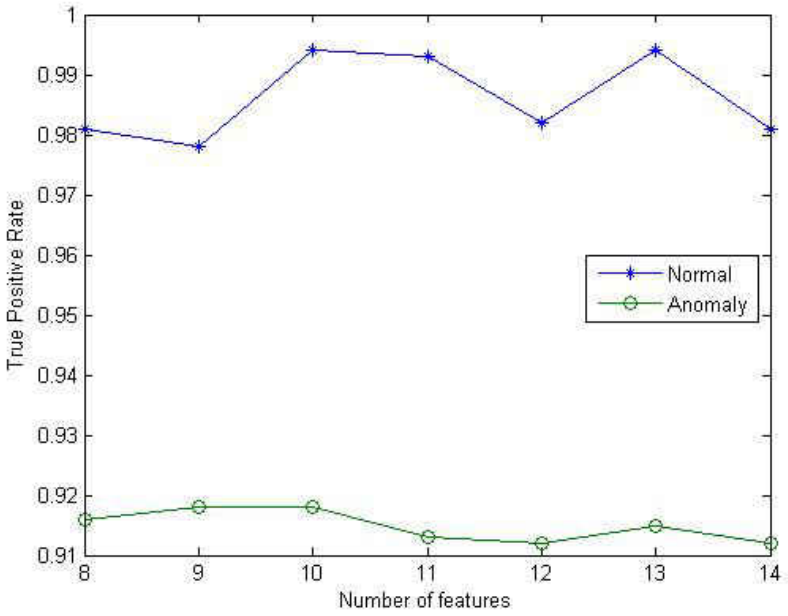

Figure 4.4: True positive rate comparison chart by different number of selected features

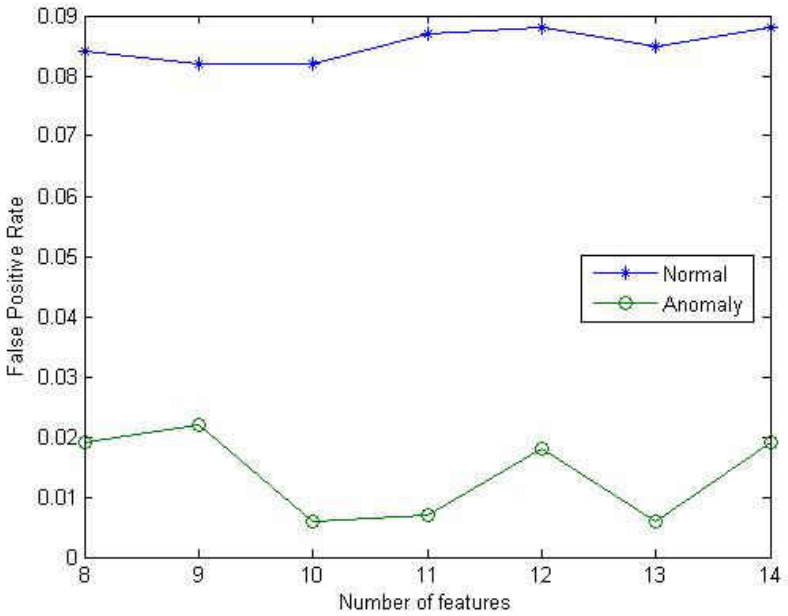

Figure 4.5: False positive rate comparison chart by different number of selected features

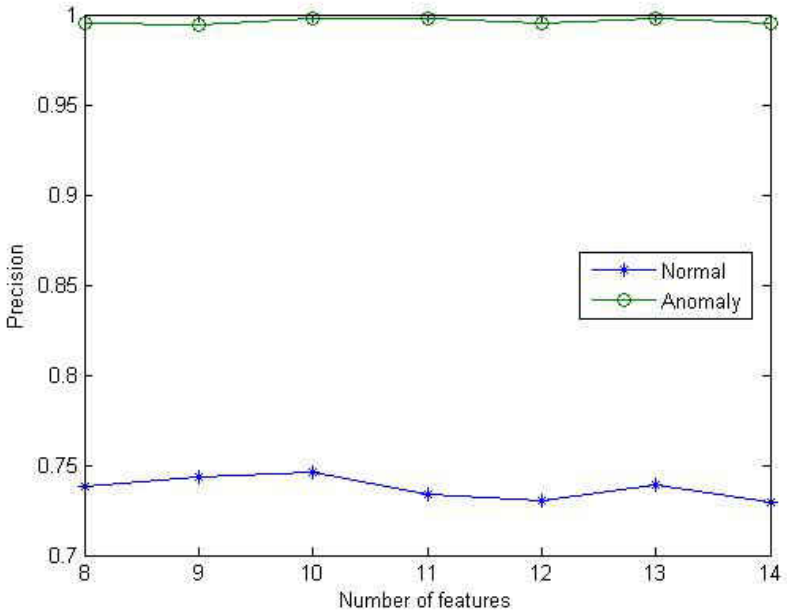

Figure 4.6: Precision comparison chart by different number of selected features

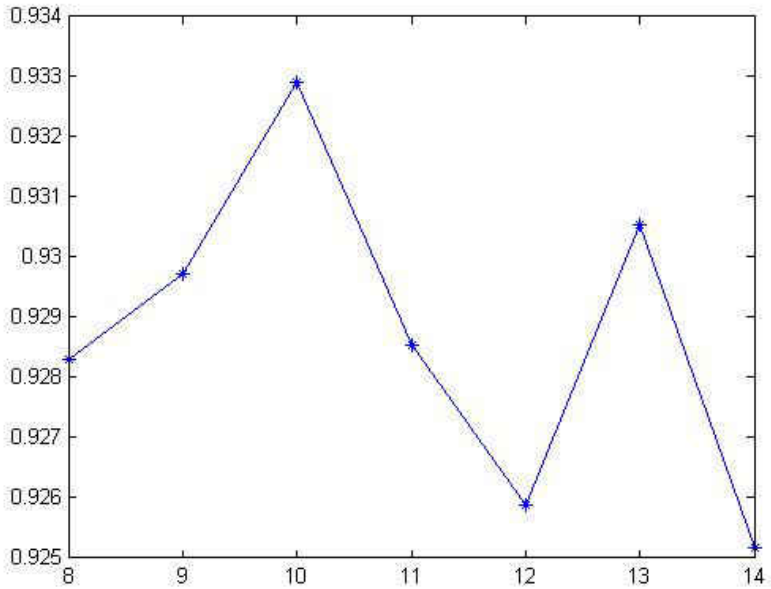

Figure 4.7: Total Accuracy comparison chart by different number of selected features

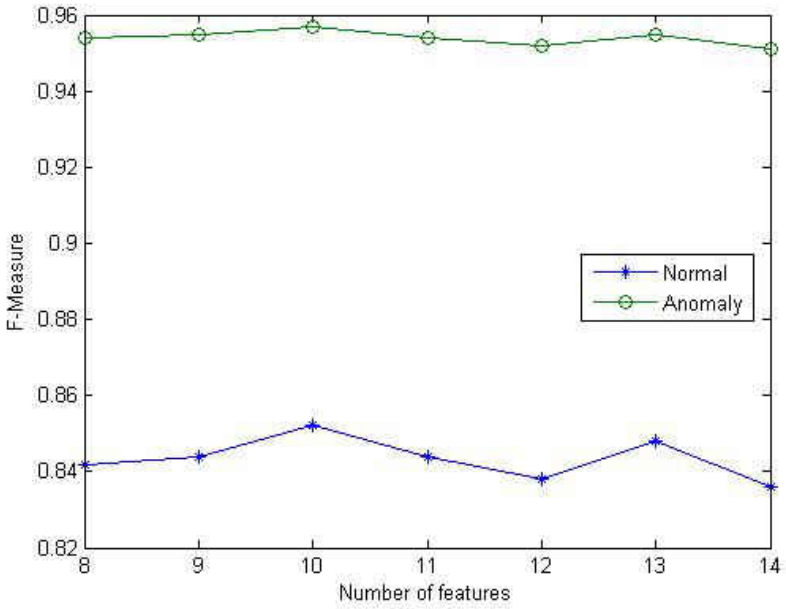

Figure 4.8: F-measure comparison chart by different number of selected features

= 0.01. Table 4.1 shows the comparison with DMIFS and FGMI on average. The first row is shown that C4.5 with all 41 features in the dataset. The second row represented DMIFS algorithm proposed by Huawen. 13 features are used by DMIFS and the performance is shown in row 2. The last two rows describe the results of the proposed algorithm FGMI. 13 features and 10 features are used to test by C4.5 respectively. And it is shown from the results that the proposed algorithm could improve the performance of all the measures. Table 4.1 highlighted in bold indicates statistically superior results in comparison to the rest.

Table 4.1: Comparison Results Between DMIFS and FGMI

Algorithm	TP Rate	FP Rate	Precision	F-Measure	Class
C4.5	**0.994**	0.09	0.728	0.841	Normal
	0.91	**0.006**	**0.999**	0.952	Anomaly
C4.5 + DMIFS (13)	0.993	0.086	0.736	0.846	normal
	0.914	0.007	0.998	0.954	anomaly
C4.5 + FGMI (13)	**0.994**	0.085	0.739	0.848	normal
	0.915	**0.006**	0.998	0.955	anomaly
C4.5 + FGMI (10)	**0.994**	**0.082**	**0.746**	**0.852**	normal
	0.918	**0.006**	0.998	**0.957**	anomaly

Another 3 algorithms were used to compare beside C4.5, and table 4.2 shows

the comparisons by the 3 different classification algorithms. The comparisons are between 41 features and 10 features which are got from the proposed algorithm. The results show that the proposed algorithm could achieve better performance, especially on F-Measure.

Table 4.2: Comparison Results by Different Classification Algorithms

Algorithm	TP Rate	FP Rate	Precision	F-Measure	Class
PART	0.994	0.087	0.733	0.844	Normal
	0.913	0.006	0.999	0.954	Anomaly
PART (10)	0.982	0.076	0.757	0.855	normal
	0.924	0.018	0.995	0.958	anomaly
Bayes	0.976	0.1	0.702	0.817	normal
	0.9	0.024	0.994	0.944	anomaly
Bayes (10)	0.979	0.1	0.702	0.818	normal
	0.9	0.021	0.994	0.945	anomaly
JRip	0.994	0.087	0.734	0.845	Normal
	0.913	0.006	0.998	0.954	Anomaly
JRip (10)	0.982	0.086	0.733	0.84	Normal
	0.914	0.018	0.995	0.953	Anomaly

One of the advantages of the feature selection method using on KDD 99 dataset is saving computation time. More features means more computation time. Figure 4.9 shows the time taken to build model of C4.5 algorithm by different number of features.

4.3.2 Results by FGMI-AHC

Table 4.3 shows comparison results by different feature selection methods using 13 selected features. The first algorithm C4.5 used 41 features to do the classification. DMIFS is dynamic mutual information feature selection method proposed by Huawen Liu, and it is introduced in chapter 3 in detailed. FGMI and FGMI-AHC are presented in section 4.1.3 and 4.2 respectively. We can see from the comparison that FGMI-AHC algorithm produces better performance on F-measure and achieves good performance on other measures.

Table 4.4 describes comparison results by different feature selection methods using 10 selected features. C4.5, DMIFS, FGMI and FGMI-AHC have the meaning as table 4.3. MMIFS is modified mutual information feature selection method raised by

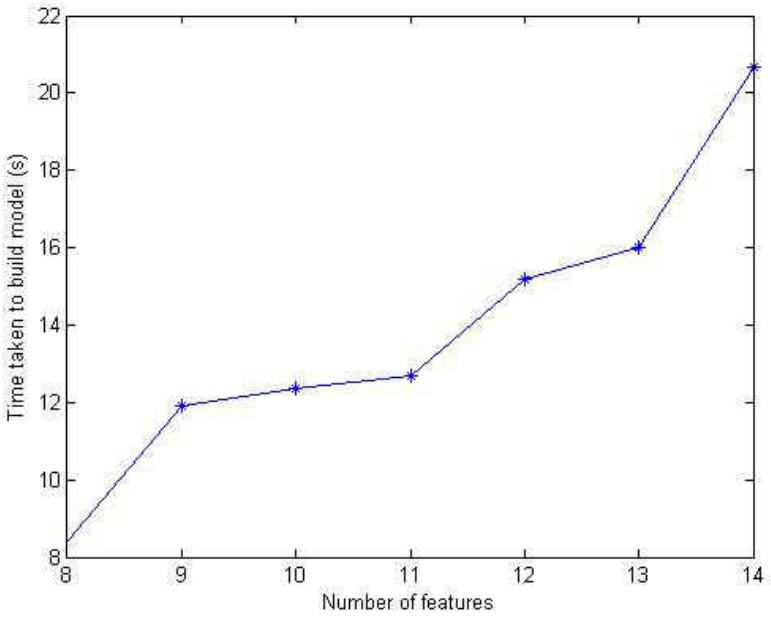

Figure 4.9: Time taken to build model comparison chart by different number of features

Table 4.3: Comparison results by different algorithms using 13 selected features

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
C4.5(41)	**0.994**	0.09	0.728	**0.994**	0.841	Normal
	0.91	**0.006**	**0.999**	0.91	0.952	Anomaly
DMIFS	0.993	0.086	0.736	0.993	0.846	normal
	0.914	0.007	0.998	0.914	0.954	anomaly
FGMI	**0.994**	0.085	0.739	**0.994**	0.848	normal
	0.915	**0.006**	0.998	0.915	0.955	anomaly
FGMI-AHC	0.993	**0.077**	**0.757**	0.993	**0.849**	normal
	0.923	0.007	0.998	**0.923**	**0.959**	anomaly

Jingping in 2014. And we can see from the comparison that FGMI-AHC could get better performance nearly in all measures. The paired t-test is again employed to compare the differences between C4.5, DMIFS, FGMI against FGMI-AHC. The tables highlighted in bold indicate statistically superior results in comparison to the rest.

Table 4.4: Comparison results by different algorithms using 10 selected features

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
C4.5(41)	**0.994**	0.09	0.728	**0.994**	0.841	Normal
	0.91	**0.006**	**0.999**	0.91	0.952	Anomaly
DMIFS	0.99	0.084	0.741	0.99	0.848	normal
	0.916	0.01	0.997	0.916	0.955	anomaly
MMIFS	0.993	0.086	0.736	0.993	0.846	normal
	0.914	0.007	0.998	0.914	0.954	anomaly
FGMI	**0.994**	0.082	0.746	**0.994**	0.852	normal
	0.918	**0.006**	0.998	0.918	0.957	anomaly
FGMI-AHC	**0.994**	**0.08**	**0.751**	**0.994**	**0.856**	normal
	0.92	**0.006**	0.998	**0.92**	**0.958**	anomaly

The purpose of comparison in table 4.3 and table 4.4 is to compare FGMI-AHC and other algorithms by the same number of selected features. Figure 4.10 illustrates the precision comparison of FGMI-AHC by different number of features.

Figure 4.11 and figure 4.12 show the F-measure and total accuracy comparison of FGMI-AHC by different number of features respectively.

From the comparison of figure 4.10 to figure 4.12, we could see better performance could be achieved when selecting 12 features by FGMI-AHC. And table 4.5 shows detailed comparison of FGMI-AHC by different number of selected features.

We can see from table 4.5 that FGMI-AHC algorithm could get best performance by selecting 12 features. And for F-measure, both normal and anomaly could achieve highest value when using 12 selected features.

Suppose there are m instances and n features in training dataset. Both of time complexity of FGMI and FGMI-AHC are $O(mn^2)$.

From the experimental results, we can see that FGMI and FGMI-AHC could get better performance but most values are close to other algorithms. On one hand, the reason is KDD 99 dataset has large number of instance, smalll improvements of

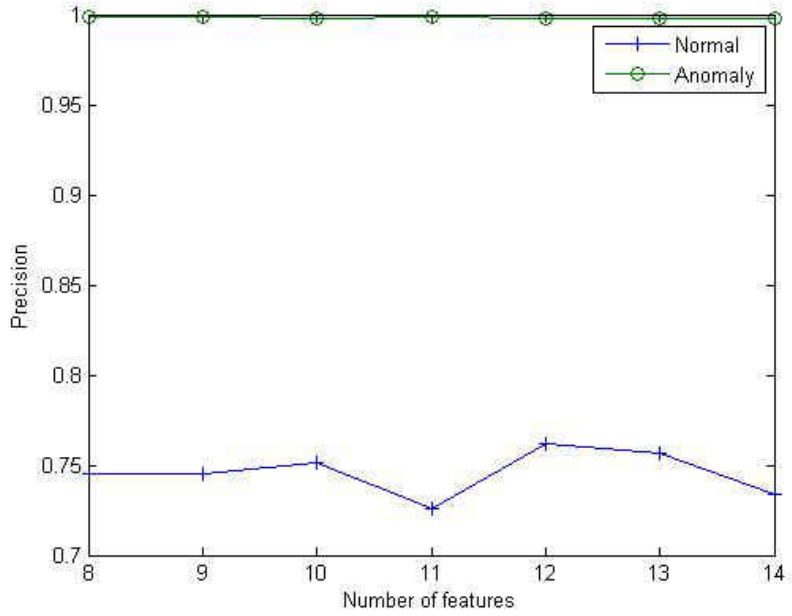

Figure 4.10: Precision comparison of FGMI-AHC by different number of selected features

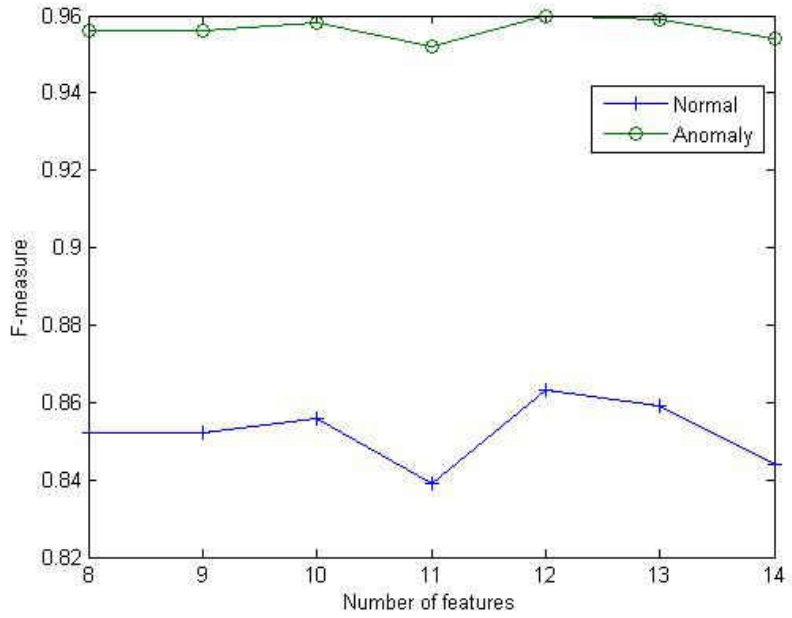

Figure 4.11: F-measure comparison of FGMI-AHC by different number of selected features

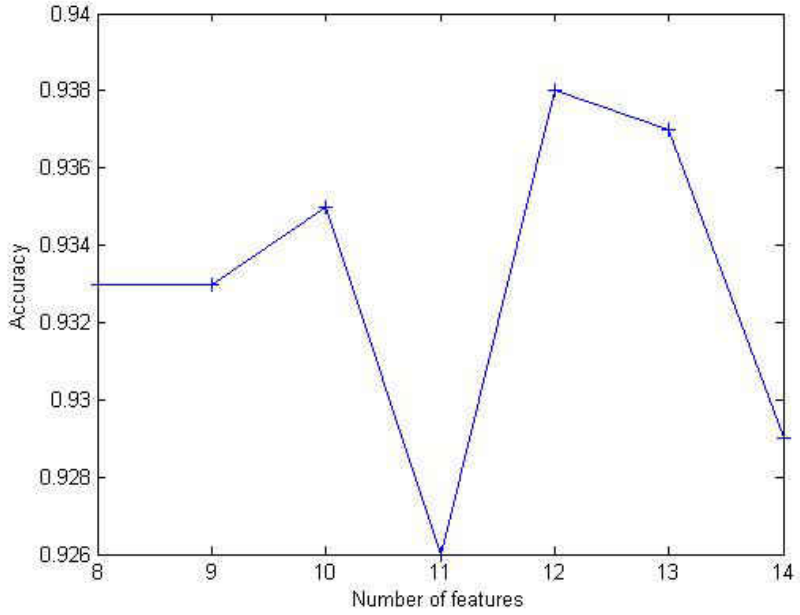

Figure 4.12: Total accuracy comparison of FGMI-AHC by different number of selected features

Table 4.5: Comparison results of FGMI-AHC by different number of selected features

No. of Features	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
8	**0.995**	0.082	0.745	**0.995**	0.852	Normal
	0.918	**0.005**	**0.999**	0.918	0.956	Anomaly
9	**0.995**	0.082	0.745	**0.995**	0.852	Normal
	0.918	**0.005**	**0.999**	0.918	0.956	Anomaly
10	0.994	0.08	0.751	0.994	0.856	normal
	0.92	0.006	0.998	0.92	0.958	anomaly
11	**0.995**	0.091	0.726	**0.995**	0.839	Normal
	0.909	**0.005**	**0.999**	0.909	0.952	Anomaly
12	0.994	**0.075**	**0.762**	0.994	**0.863**	normal
	0.925	0.006	0.998	**0.925**	**0.96**	anomaly
13	0.993	0.077	0.757	0.993	0.859	normal
	0.923	0.007	0.998	0.923	0.959	anomaly
14	0.994	0.087	0.734	0.994	0.844	normal
	0.913	0.006	0.998	0.913	0.954	anomaly

performance will bring large number of instances changed. On the other hand, the two algorithms we performed both select features first by a given number, rather than deduce to find the best feature number. Thus, we could only see the performance after feature selection process.

4.4 Summary

Section 4.2 has presented a feature grouping method based on agglomerative hierarchical clustering method. It described how to compose the group by hierarchical tree, how to get the number of groups and how to select features in each group. First of all, the mutual information between each pair of two features is calculated to be used to construct the hierarchical tree. Moreover, the proposed algorithm creates groups by a given number. Finally, the mutual information between a feature and class labels is used to select one feature in one group. Experiment results on KDD 99 dataset indicate that the proposed approach generally outperforms DMIFS, MMIFS, and FGMI algorithm. Furthermore, the comparison by different number of features shows that 12 features could get best performance indicator.

Whilst promising, the presented work opens avenues for further investigation. For instance, the mutual information between features and class labels can be used to design new algorithm. And other clustering or classification algorithms can be applied to compose groups. Moreover, more than one feature could be selected in a certain group. In future work, the proposed algorithm will be tested on other datasets and look for more effective measures or methods than mutual information theory.

Chapter 5

Online Streaming Feature Selection for IDS

In this chapter, online streaming feature selection algorithm is proposed and it is applied to the KDD99 dataset. Traditional feature selection algorithms usually need a high level of computational effort and we need to input all features at the same time and then carry out the learning process. It will consume more memory space if the dataset has more features. Online feature selection method could integrate new features as they arrive and carry out the computation. Specifically, the goal of online streaming feature selection is to develop online classifiers that involve only a small and fixed number of features for classification. This method is fit for applications where not all features could be present in advance or the feature volumns are unknown or of infinite size.

Online streaming feature selection is fit to deal with sequential training data of high dimensionality such as online intrusion detection system[32, 182]. The major contribution of this chapter is that I proposed a novel algorithm to solve real-world problems in intrusion detection system. And this online streaming feature selection algorithm could apply on other datasets as well. The application of online streaming feature selection algorithm to other datasets will be specifically described in chapter 7.

5.1 Framework for Feature Selection with Streaming Feature

Unlike the existing studies on feature selection, online streaming feature selection aims to solve the feature selection problem by online learning approach. Streaming features are features in dataset which flow one by one over time without changing the number of training samples. Compare to traditional feature selection algorithms, there are two advantages of streaming features. One is feature dimensions could grow over time and extend to an infinite size. Another one is features can be read one by one and each feature is processed online upon its arrival.

Algorithm 5.1.1 describes the framework of online feature selection algorithm. T is a training dataset and $D(F, C)$ denotes the dataset composed by all features set F and class label C. Output BCF is abbreviation of Best Candidate Feature. At first, streaming in a new feature f. And then, if f relevant to class label C, add f to BCF. Otherwise, discard f. Furthermore, judge whether f in BCF is redundant. If yes, remove it. At last, BCF will be returned.

Algorithm 5.1.1: Online Streaming Feature Selection

Input: A training dataset $T = D(F, C)$
Output: Best Candidate Feature set BCF

```
Initialize BCF = {};
Stream in a new feature f;
if f is relevant to C then
    Add f to BCF;
else
    Discard f;
if f is redundant in BCF then
    Remove f from BCF;
return the set containing selected features: BCF.
```

Xidong's work is based on this framework and proposed OSFS and fast OSFS[182]. And he presented his own method to distinguish irrelevant and redundant features. As stated in 3.1.2, he defined strong relevant, weakly relevant, irrelevant and redundant features. These definition are based on the changes of objective function when streaming a feature. The advantage of this method is that it performs well but it takes time to deduce in feature selection progress. In other words, optimizing objective the function occupies a long time.

5.2 Online Streaming Feature Selection Algorithm

Based on the framework of online feature selection algorithm in 5.1, we proposed two online feature selection algorithms. One of them shows in algorithm 5.2.1. Input is training dataset and output is BCF. This algorithm needs user to input two parameters, relevance threshold r and mutual information threshold mi. The reason is the algorithm uses relevance and mutual information to analyse the relevance and redundancy of the streaming feature. After streaming in a new feature f, the algorithm will judge whether the relevance between f and class label C is larger than r. If yes, the feature f will be added to BCF. The redundancy analysis is based on mutual information. If the mean mutual information between f and all the other features in BCF is larger then mi, the feature will be discarded.

The advantage of this algorithm is computation time will be saved and the analysis of relevance and redundancy are easier to implement. But the weakness of this algorithm is users need to understand the relevance and mutual information between the features in dataset and class label or features. Otherwise, users could not give a reasonable value to r and mi.

Algorithm 5.2.1: Online Streaming Feature Selection Algorithm 1

Input: A training dataset $T = D(F, C)$, r, mi
Output: Best Candidate Feature set BCF

```
1  BCF = {};
2  Input relevance threshold r;
3  Input mutual information threshold mi;
4  while The total number of features in T is not reached do
5      added = 0;
6      f ← get_new_feature();
7      if (Relevance(f, C) > r) then
8          added = 1;
9          BCF = BCF ⋃ f;
10     if (added) then
11         if (∑_{f_i ∈ (BCF − f)} Mutual_Information(f, f_i) > mi) then
12             BCF = BCF − f;
13 Output BCF.
```

The process of feature selection in this algorithm is only based on the relationship of features and class labels and there is no objective function. The relationships we used in this algorithm are relevance and mutual information.

Another algorithm is shown in algorithm 5.2.2. At first, the first feature will added to BCF, and users need to input a desired number K of features to be selected. If the length of BCF is smaller than K, the streaming feature is added to BCF. Else it will calculate mean relevance between all features in BCF and class label. S denotes the number of features in BCF. Then, the redundancy analysis is calculated by $redundancy = \frac{1}{S} * MI(f,C) \sum_{f_j \in S} MI(f,f_j)$. And this formula means it calculates mean mutual information between streaming feature f and all the other features in BCF, and it is weighted by mutual information between f and class label C.

Algorithm 5.2.2: Online Streaming Feature Selection Algorithm 2

Input: A training dataset $T = D(F,C)$, K
Output: Best Candidate Feature set BCF

```
BCF = {};
BCF ← get_first_feature();
K ← Input_Desired_FeatureNo;
while The total number of features in T is not reached do
    f ← get_new_feature();
    if length(BCF) < K then
        BCF = BCF ⋃ f;
    else
        BCF = BCF ⋃ f;
        relevance = 1/S Σ_{f_i∈S} Relevance(f_i, C)
        redundancy = 1/S * MI(f,C) Σ_{f_j∈S} MI(f, f_j)
        criterion = relevance − redundancy
        if criterion < (criterion|BCF - f) then
            BCF = BCF − f
Output BCF.
```

Compared to algorithm 5.2.1, this algorithm need not to input thresholds before it starts. It takes relevance and redundancy as a criterion and uses it to judge whether to select a streaming feature or not. And criterion idea is taken from [148] and we improved it. The improvement is the method of calculating relevance and redundancy. The relevance here is decided by the relevance between feature and class labels. And redundancy is calculated by mutual information, denoted as $redundancy = \frac{1}{S} * MI(f,C) \sum_{f_j \in S} MI(f,f_j)$.

5.3 Experimental Evaluation - Comparison with Traditional Feature Selection algorithms

In the following subsection, C4.5 is used to classify the dataset and compare the performance between OSFS and other algorithms proposed in other chapters. The experiments were conducted by using the KDD 99 dataset and performed on a Windows machine having configuration and Intel (R) Core (TM) i5-4308U CPU@ 2.8GHz, 2.8 GHz, 8GB of RAM, the operating system is Microsoft Windows 7 Professional. We have used an open source machine learning framework Weka 3.6.0. We have used this tool for performance comparison of our algorithm with other classification algorithms.

5.3.1 Results of OSFS1

Statistical paired t-test (per fold) is carried out to justify the significance of differences in performance of different methods, with threshold p = 0.01. The experiment is executed 10 times and table 5.1 shows the performance comparison of OSFS1 and DMIFS and FGMI on average. The table highlighted in bold indicates statistically superior results in comparison to the rest. We use OSFS1 to select 15 features and we could see from the comparison that OSFS1 could have the best performance.

Table 5.1: Comparison Results Between OSFS1 and DMIFS and FGMI

Algorithm	TP Rate	FP Rate	Precision	F-Measure	Class
C4.5	**0.994**	0.09	0.728	0.841	Normal
	0.91	**0.006**	**0.999**	0.952	Anomaly
C4.5 + DMIFS (13)	0.993	0.086	0.736	0.846	normal
	0.914	0.007	0.998	0.954	anomaly
C4.5 + FGMI (13)	**0.994**	0.085	0.739	0.848	normal
	0.915	**0.006**	0.998	0.955	anomaly
C4.5 + OSFS1 (15)	**0.994**	**0.083**	**0.743**	**0.85**	normal
	0.917	**0.006**	0.998	**0.956**	anomaly

In this algorithm, the number of selected features is decided by the input thresholds r and mi. Regarding this experiment, we set $r = 0.2$ and $mi = 0.05$. The two values could be decided by experience or understanding of the dataset or according to the largest relevance and mutual information between features and class labels. The number of selected features will be changed if you change the input value r and mi.

5.3.2 Performance evaluation of OSFS2

We need to select some features to do the performance comparison for algorithm OSFS2 since the algorithm requires to input a number that user want to select the feature number. As stated in 4.3.1, we select 8 to 14 features to do performance evaluation. Figure 5.1, 5.2, 5.3, 5.4, 5.5 show true positive rate, false positive rate, precision, F_Measure and accuracy comparison by different number of selected features respectively. From the comparison we could see that we could get best performance when we select 9 features by using algorithm OSFS2. Table 5.2 describes comparison of OSFS2 and other algorithms using 10 features. And from the table, we could see that OSFS2 could achieve better performance, especially F-Measure. The paired t-test is again employed to compare the differences in performance of different methods. The table highlighted in bold indicates statistically superior results in comparison to the rest.

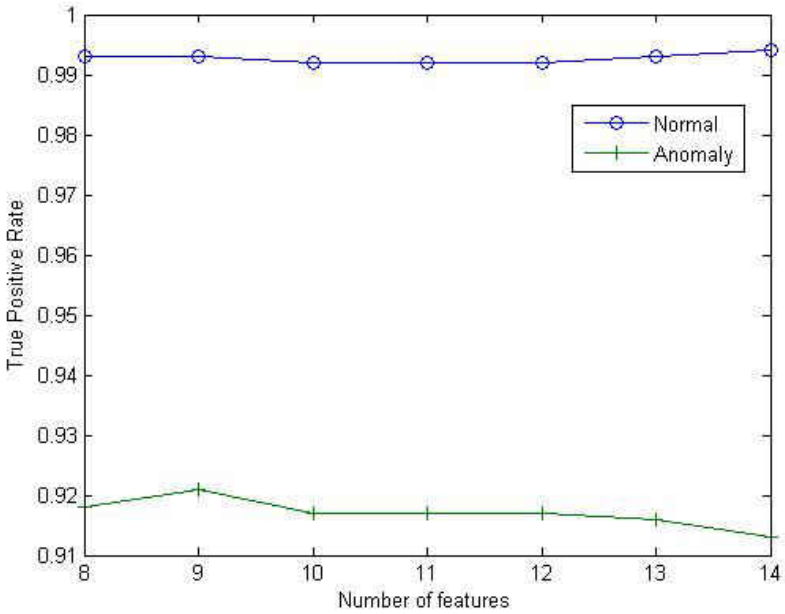

Figure 5.1: True positive rate comparison chart by different number of selected features

Algorithm OSFS2 will vary depending on the order in which features are streamed. The results in table 5.2 used the original order of the dataset. Next, we change the order of streaming feature randomly and get its performance. In this test, we did this experiment 10 times and selected 9 features every time. The average performance

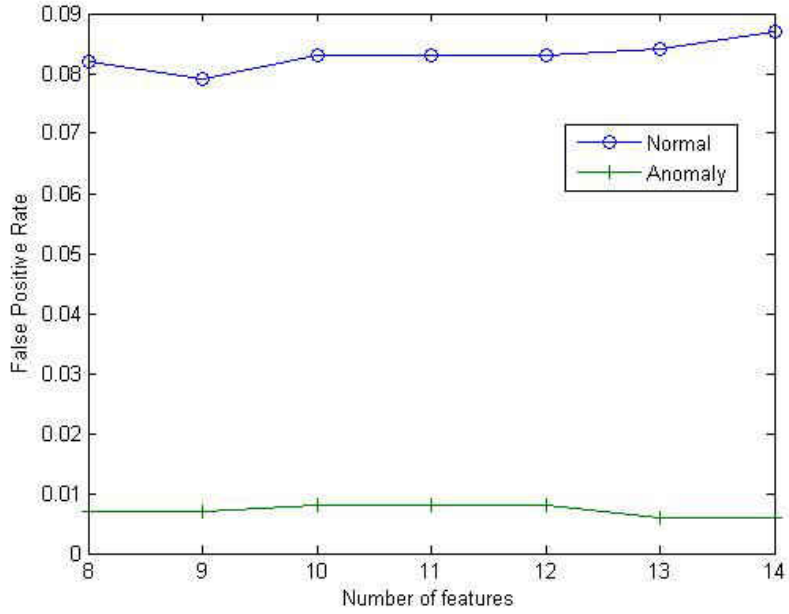

Figure 5.2: False positive rate comparison chart by different number of selected features

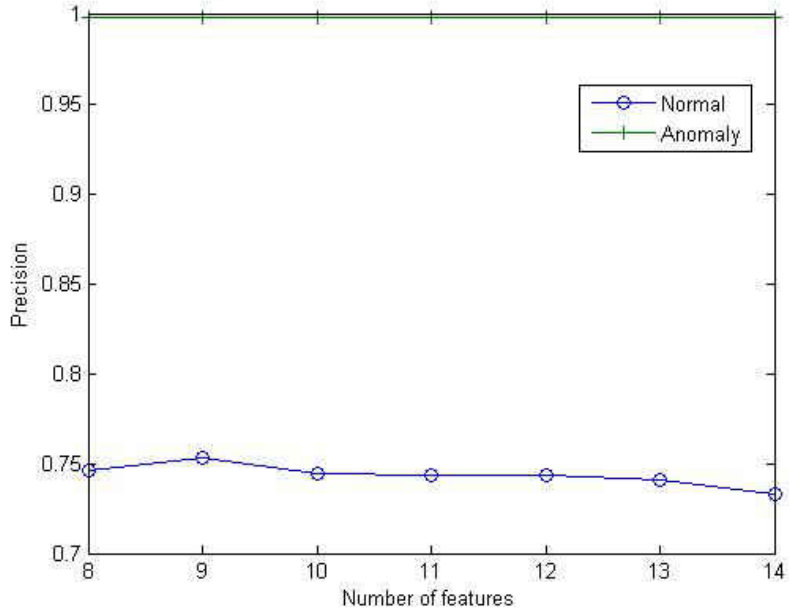

Figure 5.3: Precision comparison chart by different number of selected features

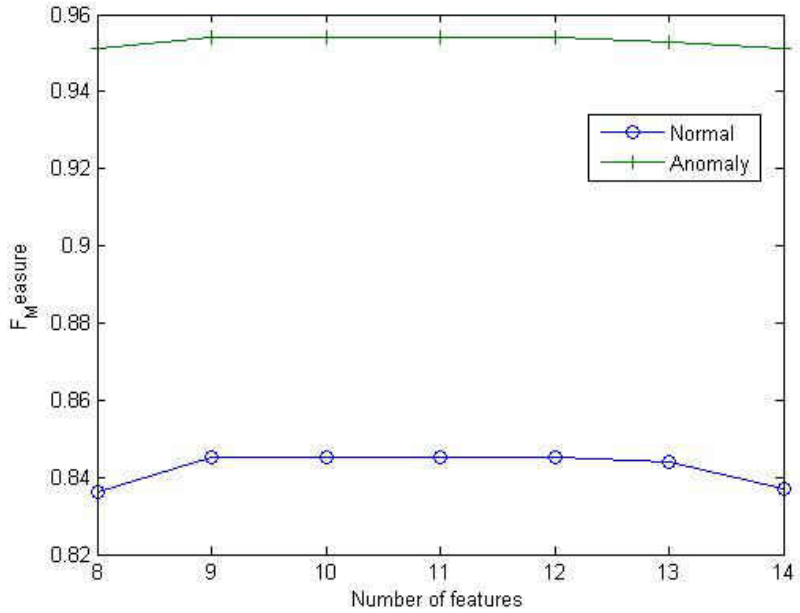

Figure 5.4: F_Measure comparison chart by different number of selected features

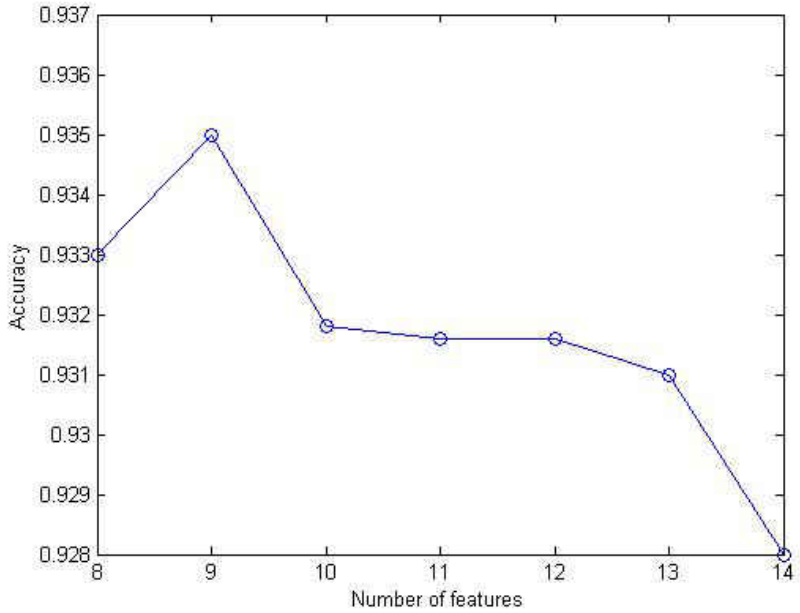

Figure 5.5: Accuracy comparison chart by different number of selected features

Table 5.2: Comparison Results Between OSFS2 and other algorithms

Algorithm	TP Rate	FP Rate	Precision	Recall	F-Measure	Class
C4.5(41)	**0.994**	0.09	0.728	**0.994**	0.841	Normal
	0.91	**0.006**	**0.999**	0.91	0.952	Anomaly
DMIFS(10)	0.99	0.084	0.741	0.99	0.848	normal
	0.916	0.01	0.997	0.916	0.955	anomaly
MMIFS(10)	0.993	0.086	0.736	0.993	0.846	normal
	0.914	0.007	0.998	0.914	0.954	anomaly
FGMI(10)	**0.994**	0.082	0.746	**0.994**	0.852	normal
	0.918	**0.006**	0.998	0.918	0.957	anomaly
FGMI-AHC(10)	**0.994**	0.08	0.751	**0.994**	**0.856**	normal
	0.92	**0.006**	0.998	0.92	**0.958**	anomaly
C4.5 + OSFS2 (9)	0.993	**0.079**	**0.753**	0.993	**0.856**	normal
	0.921	0.007	0.998	**0.921**	**0.958**	anomaly

of this 10 test is shown in table 5.3. And precision and F-measure are shown in figure 5.6 and figure 5.7. From the test, we can see that streaming order affects performance of results. And we also found that we could get better performance if we stream features which have larger relevance with class labels.

Table 5.3: Average performance of different feature steaming order

TP Rate	FP Rate	Precision	F-Measure	Class
0.988	0.088	0.732	0.841	Normal
0.912	0.012	0.997	0.953	Anomaly

Suppose there are m instances and n features in training dataset. The time complexity of OSFS1 and OSFS2 are $O(mn)$ and $O(mn^2)$ respectively.

5.4 Summary

In this chapter, I introduced two online feature selection algorithms. And from the comparison with other feature selection algorithm we proposed before, we could see that OSFS algorithms could get better performance. But there are some disadvantages in the two algorithms, such as we need to input some threshold or desired number of features you will select from datasets. And they need to be improved by revising some part of the algorithms. Such as we might deeply analyse the relationship of relevance and redundancy of features in a dataset and design an algorithm that removes the need for any threshold or desired number.

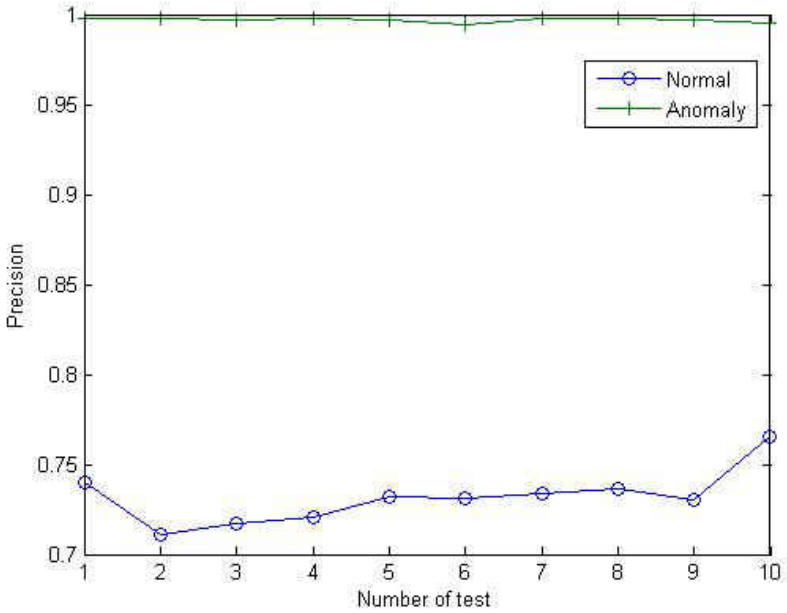

Figure 5.6: Precision of different feature streaming order test

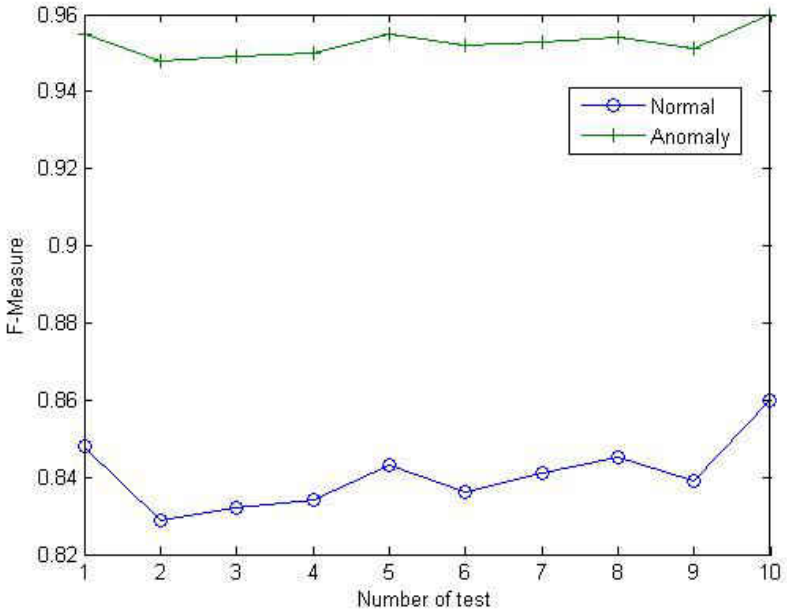

Figure 5.7: F_Measure of different feature streaming order test

Chapter 6

Unsupervised Network Intrusion Detection System

MOST current network intrusion detection systems employ signature-based methods or data mining-based methods which rely on labelled training data[187, 49]. This training data is typically expensive to produce. Moreover, these methods have diffculty in detecting new types of attack[162, 21]. Using unsupervised anomaly detection techniques, however, the system can be trained with unlabelled data and is capable of detecting previously unseen attacks[27]. In this chapter, two algorithms are proposed. Cascading fuzzy C means clustering and C4.5 decision tree classification algorithm combine an unsupervised method with a supervised method. Cluster accumulation ensemble with hierarchical clustering algorithm is an unsupervised algorithm. Both of them are evaluated on KDD 99 dataset.

6.1 Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm

We implemented a scheme to find anomaly instances using fuzzy C means clustering and C4.5 decision tree algorithm. We regard all the attack instances as anomaly instances and so convert a multiple class classification problem to a binary class classification. Figure 6.1 shows the flow chart of proposed scheme. First of all, we use feature selection methods on training data to get some selected features. But

the features have different kinds of data structure. For this reason, we normalize the data so that all attribute values are between 0 and 1. Then, we use fuzzy C-means method to divide the training data into two clusters and get two centres. As presented in 4.1.3, fuzzy C-means is a clustering method by using membership function. Moreover, we calculate the membership function between each test data instance and each cluster. The test data instance is allocated to the cluster which has higher membership. Finally, fuzzy c-means is cascaded with the C4.5 by building decision trees using the instances in each cluster. We used C4.5 algorithm to classify the test data as an anomaly or a normal instance. Cascading could solve a problem when most of instances from one class and very few instances from other classes are in a single cluster. Such clusters, which are dominated by a single class, show weak association to other classes. In KDD 99 dataset, most of the instances are DOS attacks, and cascading two machine learning methods could get better results. Each part of the process is now described in greater detail.

6.1.1 Application to Feature Selection

Table 6.1 shows four feature selection tests on KDD99 training dataset and we select 6 to 8 features. The results show that most of the features selected by different feature selection methods are the same. Such as, all 6 features in test 3 are in the results of test 1 and six features in the result of test 4 are in test 1 results. We used the 8 features selected by test 1. They are protocol_type, service, flag, src_bytes, dst_bytes, count, diff_srv_rate, dst_host_same_src_port_rate. The evaluator in test 1 is based on Correlation-based Feature Selection, which is one of the most well-known and used filters, and ranks feature subsets according to a correlation based heuristic evaluation function. And one advantage of feature selection is gaining speed.

Table 6.1: Results obtained by four feature selection methods over KDD99 training dataset

Test No.	Search Method	Attribute Evaluator	No. of Selected Features
1	BestFirst	CfsSubsetEval	8
2	Ranker	ConsistencySubsetEval	7
3	FCBFSearch	SymmetricalUncertAttributeSetEval	6
4	Randomsearch	AttributeSubsetEvaluator	7

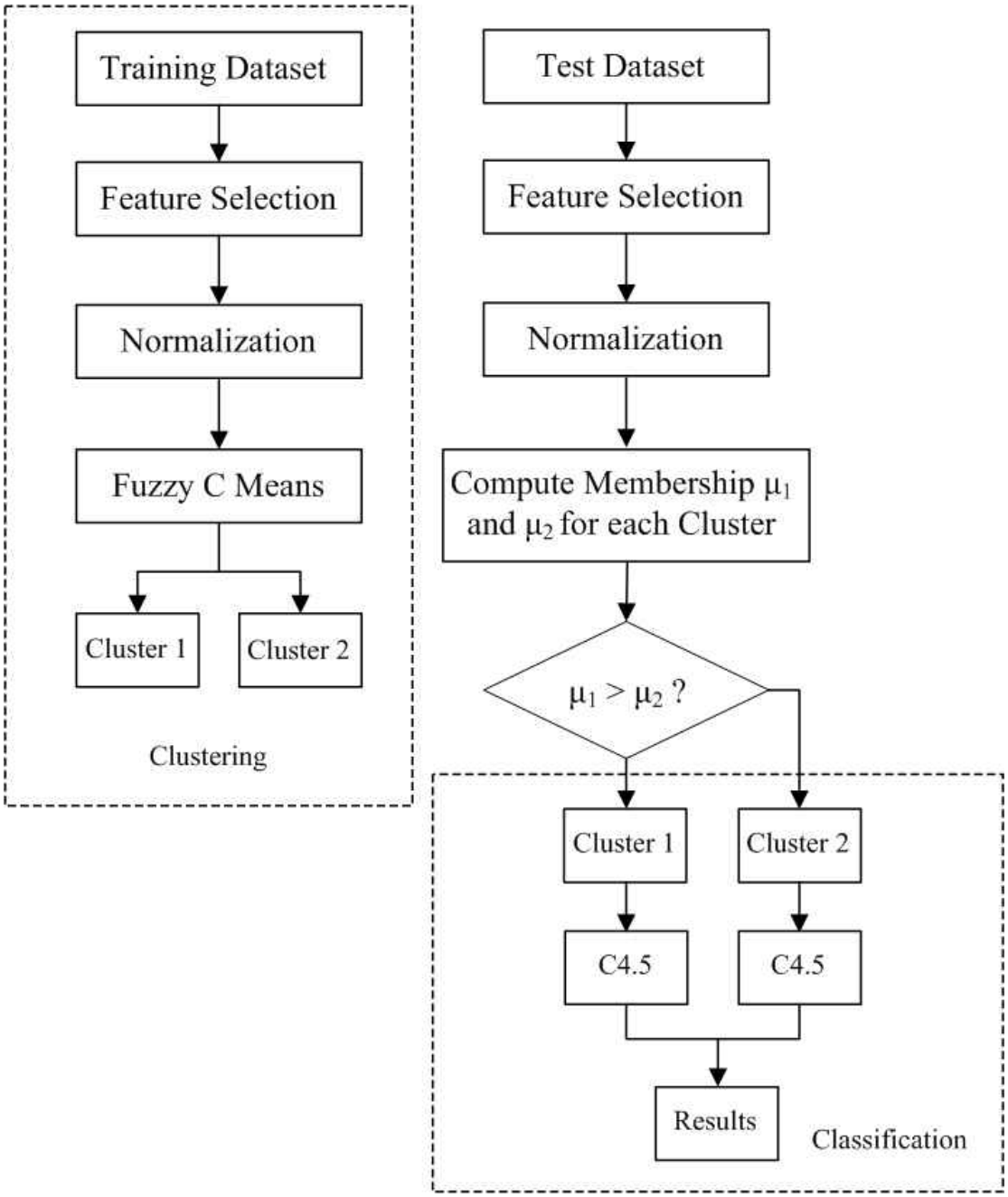

Figure 6.1: Flow chart of proposed scheme

6.1.2 Normalization

The 8 features we used have two types. The protocol_type, service, flag are symbolic and the other five features are continuous. The protocol_type has 3 values, service has 66 values, and flag has 11 values. Table 6.2 shows the minimum and maximum value of each feature, as well as its mean, standard deviation and the number of distinct examples of the five continuous features.

Table 6.2: Unbalanced continuous features of KDD Cup 99 dataset

Feature	Max.	Min.	Mean	StdDev	Distinct
src_bytes	693375640	0	3025.61	988218.1	3300
dst_bytes	5155468	0	868.5324	33040	10725
count	511	0	332.2857	213.1474	490
diff_srv_rate	1	0	0.020982	0.082205	78
dst_host_same_src_port_rate	1	0	0.601935	0.481309	101

Normalization converts all the data in the dataset between 0 and 1. For a particular continuous data x_i, normalization follows equation 6.1,

$$Normalized(x_i) = (x_i - X_{min})/(X_{max} - X_{min}) \tag{6.1}$$

where X_{min} is the minimum value for variable X, X_{max} is the maximum value for variable X. For a specific symbolic feature, we assigned a discrete integer to each value and then used equation 6.1 to normalize it.

6.1.3 Implemented Algorithm

In this section, we will show the algorithm of the fuzzy C means clustering cascade with C4.5 decision tree classification methods for supervised anomaly detection. Fuzzy C means allocate data points to clusters is not “hard” (all-or-nothing) but “fuzzy” in the same sense as fuzzy logic. C4.5 is a well-known classification algorithm used to generate a decision tree. We use fuzzy C means algorithm to group 2 clusters and we get 2 centers. We then used C4.5 to classify in each cluster. Algorithm 6.1.1 is shown as follows.

At first, the algorithm initialise U which denotes the membership matrix of D_{train}, in other words, sets $U^{(0)}$ to U. And U_{ij} is degree of membership of x_i in cluster j. x_i is the ith of instance and c_j is the jth center of cluster. Then, it uses fuzzy

Algorithm 6.1.1: Cascading Fuzzy C Means clustering and C4.5 decision tree classification algorithm

Input: $D_{train}, D_{test}(z_i \in D_{test})$
Output: Classified test instance z_i to normal or anomaly

```
Initialise membership matrix of D_train, U = [u_ij] ← U^(0)
while ||U^(k+1) − U^(k)|| ≥ ε do
    Calculate the centres C_1,C_2;
    Update U^(k), U^(k+1);
for z_i ∈ D_test do
    Compute u_ij, j = 1,2;
    Find Higher Membership to z_i;
    Assign z_i to Higher Membership Cluster;
Classify each cluster by C4.5;
Return C_1, C_2 and clusters.
```

c-means algorithm to calculate centers and membership matrix. And c_j could be calculated by 6.2, where m is any real number greater than 1. Membership matrix is calculated by 6.3, where k are the iteration steps and the iteration will stop when $||U^{(k+1)}-U^{(k)}|| \geq \epsilon$, whereas ϵ is a termination criterion between 0 and 1. In this case, there are two centers C_1 and C_2. Furthermore, for each instance in test dataset D_{test}, calculate membership value to each center. After that, it finds higher membership and assigns the instance to higher membership cluster. In other words, the instance in test dataset will divided into two clusters according to the degree of membership to C_1 and C_2 in this case.

$$C_j = \frac{\sum_{i=1}^{n} u_{ij}^m \cdot x_i}{\sum_{i=1}^{n} u_{ij}^m} \tag{6.2}$$

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{||x_i - c_j||}{||x_i - c_k||} \right)^{\frac{2}{m-1}}} \tag{6.3}$$

This algorithm has a limitation that users need to predefine the number of clusters. In this case, all attacks are seen as abnormal data and there are two clusters, normal and anomaly. Clusters could show internal structure of data and could help to find unknown attacks. But for multi-class scenario, if users do not know the number of attcks in test dataset, the algorithm lose effectiveness.

6.2 New Evidence Accumulation Clustering with Hierarchical Clustering Algorithm

Evidence accumulation clustering method combine the results of multiple clusterings into a single data partition, by viewing each clustering result as an independent evidence of data organization. It is can be presented as follows. Let $X = \{x_1, x_2, ..., x_N\}$ be a dataset and $\mathscr{E} = \{E_1, E_2, ..., E_M\}$ be M ensemble members. For X, each ensemble returns a set of clusters $E_m = \{C_1^m, C_2^m, ..., C_{K_m}^m\}$, where K_m is the number of clusters constructed by E_m. So $\bigcup_{k=1}^{K_m} C_k^m = X$, and $\{C_1, C_2, ..., C_n\} = \bigcup_{m=1}^{M} E_M$ denotes that the

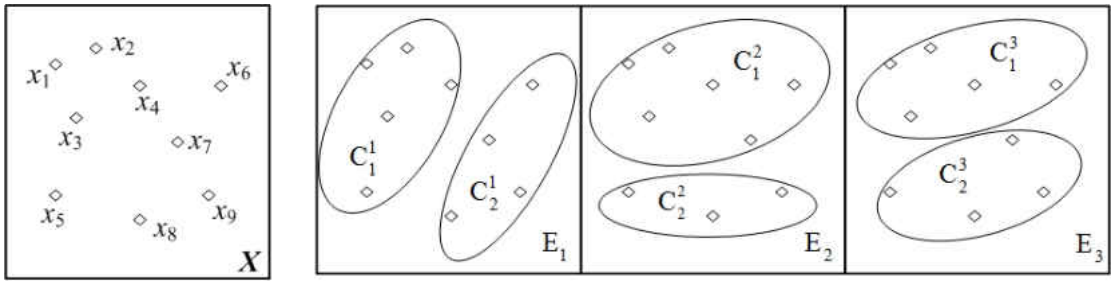

Figure 6.2: Example of Cluster Ensemble

clusters generated by all ensemble members together form a set of base clusters for ensemble, where $n = \sum_{m=1}^{M} K_m$. There are two procedures of cluster ensemble. First, ensemble members are generated. Second, a consensus function is then applied on those ensemble members to generate the final clustering result.

Most evidence accumulation clustering algorithms use all features in dataset to do clustering. And it usually costs large computation time on clustering if it is tested on a large dataset. We proposed a new algorithm and it will cluster by each feature rather than all the features in the same time. Figure 6.3 shows schematic diagram of the proposed method. $F_1, F_2, ...F_M$ denote M features in dataset X and each E_m is calculated according to F_m. That means the number of ensembles is the number of features in dataset X. And the algorithm is shown in algorithm 6.2.1.

From the algorithm description, we can see that it deals with features of the dataset one by one. In other words, it clusters features one by one rather than clustering all features at the same time. At first, the algorithm gets the instance number and feature number to n and d and sets co_assoc to a null $n \times n$ matrix. Co_assoc denotes co_association matrix. Then, it gets a feature from the dataset and

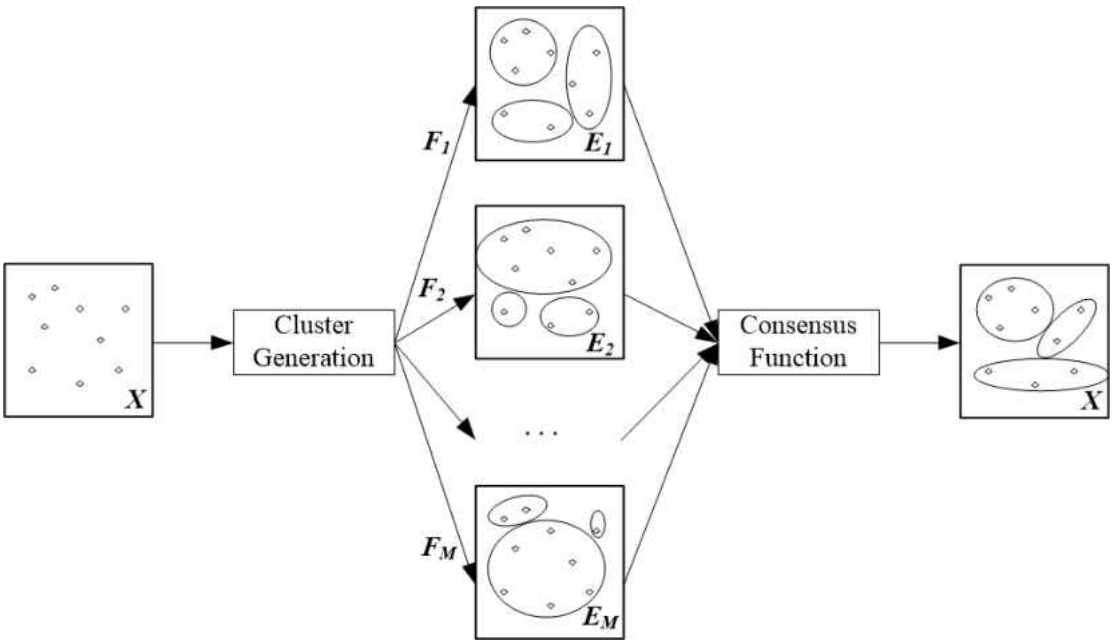

Figure 6.3: Schematic diagram of the proposed method

Algorithm 6.2.1: New Evidence Accumulation Clustering Algorithm

Input: A training dataset $T = D(F, C)$
Output: Combined data partition E

$n \leftarrow$ Get instance number of dataset T;
$d \leftarrow$ Get feature number of dataset T;
Set co_assoc to a null $n \times n$ matrix;
for $i = 1$ **to** d **do**
    $data\_column \leftarrow$ Get the i^{th} feature from dataset T;
    Run clustering algorithm on $data\_column$ and produce a partition E_i;
    Update the co_association matrix: for each pattern pair (i, j) in the same cluster in E_i, set $co\_assoc(i, j) = co\_assoc(i, j) + 1/d$;
Compute the SL dendrogram of co_assoc and identify the final patition E as the ones with the highest lifetime;
return Combined data partition E.

runs the clustering algorithm on this feature and produce a patition. Moreover, it updates the co_association matrix. For each pattern pair (i, j) in the same cluster in E_i, set $co\_assoc(i, j) = co\_assoc(i, j) + 1/d$. The number of clustering is d. Finally, we use hierarchical clustering algorithm with single link to get the final partition.

As stated above, the number of ensembles is fixed as the number of features if a dataset is given. And the cluster generation process only use one feature rather

than all features in dataset. Thus, computation time will be saved compare to the similar algorithms which use all features to get ensembles. After d steps of iteration, we get co_association matrix and then we use hierachical clustering methods on it. And SL denotes single linkage which is used to describe the distance of clusters in hierachical clustering algorithm.

6.3 Experimental Results

6.3.1 Measures of Performance Evaluation

Our proposed scheme is conducted by six measures which is presented in 3.3.1. In our proposed scheme, we use the training dataset to construct the decision tree model and then reevaluate on the test dataset and get TP, FP, TN, FN in each cluster. After that, we calculate the four values for the test dataset. And finally, we calculate the measures for the test dataset.

6.3.2 Results by Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm

Table 6.3 describes the performance evaluation comparison of C4.5, C4.5 with feature selection and the proposed scheme on the KDD Cup 99 test dataset. The total accuracy of these three algorithms are 0.926, 0.931, 0.935 respectively. And the comparison shows that most precision, recall and F-measure results are improved by using the proposed scheme. From the comparison on the table 6.3, we can see that

Table 6.3: Performance evaluation comparison

Algorithm	TPR	FPR	Precision	Recall	F-Measure	Class
C4.5	**0.994**	0.090	0.728	**0.994**	0.841	Normal
	0.910	**0.006**	**0.999**	0.910	0.952	Anomaly
C4.5 with FS	0.990	0.084	0.741	0.990	0.847	normal
	0.916	0.010	0.997	0.916	0.955	anomaly
Proposed Scheme	0.990	**0.079**	**0.753**	0.990	**0.855**	Normal
	0.921	0.010	0.998	**0.921**	**0.958**	Anomaly

the proposed algorithm could achieve better performance, especially on the measure of precision, accuracy and F-measure. Suppose there are m instances and n features in training dataset. The time complexity of proposed algorithm is $O(2mnt)$, where t is the number of iterations.

6.3.3 Results by New Evidence Accumulation Clustering with Hierarchical Clustering Algorithm

We tested the proposed algorithm on KDD99 test dataset. But the instances are too large to test. The reason is that co_association matrix is $n*n$, where n is the instance number in a dataset. Ten percent of KDD99 include about 500,000 instances. It very diffcult to deal with the matrix by a normal computer. So we use resample function in weka to get 1% samples in the dataset to test proposed algorithm.

We compare our proposed algorithm with the combining multiple clusterings algorithm by Ana L.N. Fred which is proposed in 2005[47, 48]. She used whole dataset to do n times clustering and then update the co_association matrix. And she also used SL dendrogram to deal with co_association matrix. And the dendrogram by this algorithm is shown in figure 6.4.

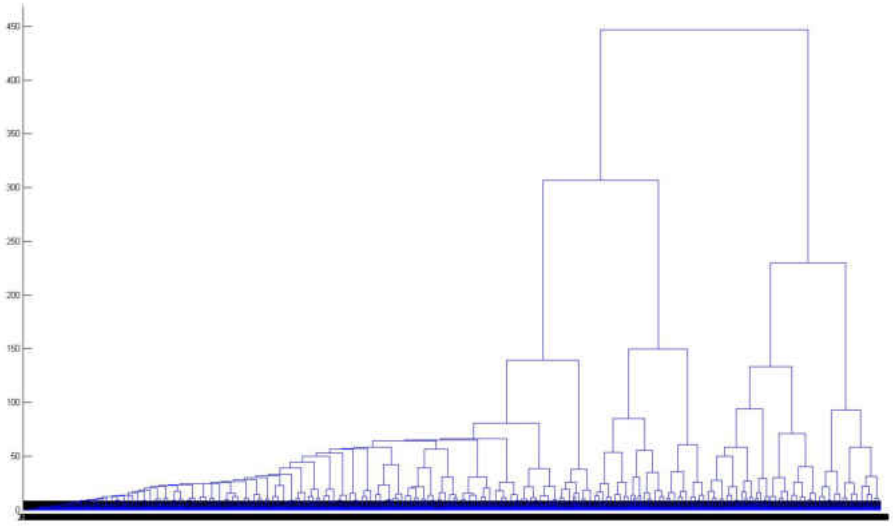

Figure 6.4: Dendrogram of Ana's algorithm

And our proposed algorithm's dendrogram is shown in figure 6.5.

We used K-means algorithm to do clustering and we compare the total accuracy by K-means, Ana's algorithm and our proposed algortihm. The results are shown in table 6.4. All the algorithms are tested 30 times and total accuracy are the average values. And from the results, we could see that our proposed algorithm has the same

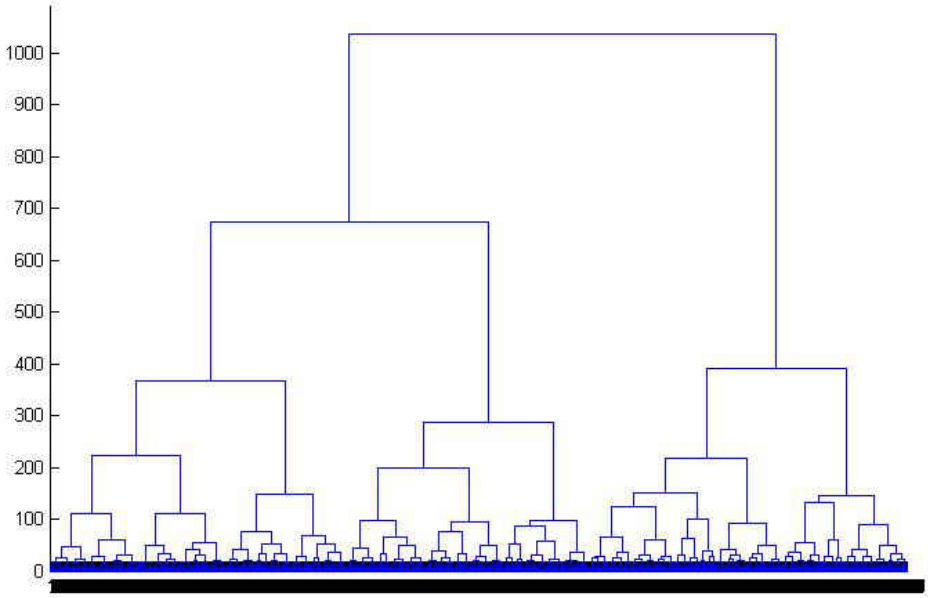

Figure 6.5: Dendrogram of proposed algorithm

performance as Ana's algorithm, and better than simple K-means algorithm. In Ana's algorithm, it runs N times clusterings. And in every time, it uses K-means algorithm on whole a dataset to produce a partition and then construct co-association matrix. Unlike runing K-means N times, the number of clustering of our proposed algorithm is depends on the number of features of datasets. The experiment is executed 10 times and the differences between Ana's algorithm and proposed algorithm are statistically evaluated using paired t-test with two-tailed $p = 0.01$. The two algorithms achieve statistically the same result. Suppose there are m instances and n features in training dataset. The time complexity of proposed algorithm is $O(m^2n)$.

Table 6.4: Comparison Results Between Proposed algorithm and Other algorithms

Algorithm	Total accuracy
K-Means	72.85%
Ana's algorithm	79.77%
Proposed algorithm	79.77%

From table 6.4 we can see that unsupervised methods like K-means could not achieve higher performance compared to supervised methods on KDD 99 dataset. Although ensemble accumulation the clusters methods could help to improve the

performance, it takes more computation time since it needs to run N times clustering by using all features. On the contrary, our proposed algorithm generate clusters by one feature every time. Although the result is the same as Ana's algorithm, the proposed algorithm greatly reduces the computational complexity. More datasets are tested and compared with Ada's algorithm in 7.3.

6.4 Summary

This chapter proposed two unsupervised algorithms. The first is based on the combination of feature selection, fuzzy C means and C4.5 algorithms that improve the performance results of classifiers while using a reduced set of features. It has been applied to the KDD Cup 99 dataset in the intrusion detection field. We used a normalization method on the KDD 99 training dataset and test dataset before applying the proposed scheme to the dataset. The method improves the performance results obtained by C4.5 while using only 19.5% of the total number of features. Performance analysis is assessed against six measures. This method gives impressive detection precision accuracy and F-measure in the experiment results. An additional advantage is memory and time costs reduction for C4.5 classifier.

The second algorithm is a new evidence accumulation clustering algorithm. It clusters features one by one. It promises to save computation time and is suitable to deal with big data. From the comparison, we could see that it can achieve better performance.

Chapter 7

Application to Other Datasets

In this chapter, we will test the described algorithms on other datasets. Since the algorithms described in chapter 3 to chapter 6 are either classification algorithms or clustering algorithms, they all could deal with datasets for classification or clustering. We will test three proposed algorithms in this chapter and compare their performance with other algorithms.

7.1 Test by FGMI-AHC

In chapter 3 and 4, three algorithms are proposed based on mutual information. They are Modified Mutual Information-based Feature Selection Algorithm(MMFS), Mutual Information-based Feature Grouping Algorithm(FGMI), Feature Grouping by Agglomerative Hierarchical Clustering Algorithm(FGMI-AHC). From the performance evaluation comparison of KDD 99 dataset, we can see that, FGMI-AHC could achieve the best performance. Thus, we test FGMI-AHC algorithm on other dataset instead of testing all the three algorithms in this chapter.

7.1.1 Dataset introduction

To verify FGMI-AHC is effective, we choose 14 datasets from UCI machine learning repository in our experiment[8]. These datasets are often used to compare algorithms performance in machine learning and data mining area. Table 7.1 shows outline descriptions of the datasets, including dataset name, instance number, feature number

and class number. Detailed information of the datasets could obtained from the UCI website. From table 7.1, we can see that these datasets from different areas, instance number from 187 to 11055, feature dimension from 9 to 279, and including two class and multi-class labels. This pluralism could help to verify performance of the algorithm in different conditions.

Table 7.1: Datasets for test in experiments

NO.	Dataset name	Instance NO.	Feature NO.	Class NO.
1	Glass	214	9	7
2	Pendigits	3498	16	10
3	Phishing Websites	11055	30	2
4	Spambase	4601	57	2
5	Ionosphere	351	34	2
6	Statlog	2000	36	6
7	Biodeg	1055	41	2
8	ThoracricSurgery	470	16	2
9	SPECT	187	22	2
10	SPECTF	187	44	2
11	Semeion	1593	265	2
12	CNAE-9	1080	256	9
13	Optdigits	1797	64	10
14	Arrhythmia	452	279	16

Since these datasets from different areas, some of them have missing values and some of them need to change format. Thus, before testing algorithms, we need to preprocess the datasets. We use mean value to fill missing values. And we change discrete string to discrete digital number since this is our proposed algorithm's requirement.

7.1.2 Results for FGMI-AHC

To compare feature selection algorithms' performance evaluation, we use four different classifiers since a classifier might prefer a specific feature selection algorithm. Table 7.2, 7.3, 7.4, 7.4 show performance by decision tree (C4.5), 1-Nearest Neighbor (1-NN), Support Vector Machines (SVM) and Naive Bayes classifiers respectively. These four classifiers represent four different machine learning algorithms and they are able to represent some feature selection algorithms. We compare our proposed algorithm FGMI-AHC with Information Gain (IG) and ReliefF which are introduced in chapter 2.

In this experiment, we use weka platform to test classification algorithms and the parameters are set to default value for each learning algorithm. Classification accuracy is usually used as performance evaluation criteria of a subset. Table 7.2, 7.3, 7.4, 7.5 show classification accuracy comparison by different learning algorithms. The number in parentheses next to each accuracy number are subset size obtained from different algorithms. Some of the datasets have training dataset and test dataset separately, some have training dataset only. In order to obtain more reliable classification performance, these tests use 10 times 10-fold cross-validation for those datasets who have training dataset only. In 10-fold cross-validation test, dataset is divided into 10 parts. And 9 parts will be used as training data in turn, one part is as the test data. Each simulation experiment will draw the appropriate classification accuracy rate, and the average of 10 times as a result of the accuracy of an algorithm.

Table 7.2: Classification accuracy comparison by C4.5

NO.	All features	FGMI-AHC	IG	ReliefF
1	65.89%(9)	**68.69%**(4)	67.29%(4)	64.49%(4)
2	92.05%(16)	**86.45%**(6)	80.07%(6)	84.85%(6)
3	92.98%(30)	**94.69%**(10)	94.37%(10)	94.56%(10)
4	79.29%(57)	**91.81%**(9)	90.78%(9)	84.39%(9)
5	82.62%(34)	92.31%(8)	**92.88%**(8)	**92.88%**(8)
6	79.6%(36)	78.95%(9)	**85.2%**(9)	59.4%(9)
7	75.92%(41)	81.33%(10)	79.81%(10)	**85.02%**(10)
8	78.51%(16)	**86.17%**(7)	85.11%(7)	85.11%(7)
9	60.96%(22)	**72.73%**(9)	70.05%(9)	70.05%(9)
10	69.52%(44)	**73.26%**(11)	72.73%(11)	69.52%(11)
11	92.09%(265)	**94.85%**(19)	94.1%(19)	94.48%(19)
12	93.15%(256)	**75.74%**(18)	75.46%(18)	69.91%(18)
13	89.43%(64)	**86.7%**(20)	83.81%(20)	86.09%(20)
14	61.73%(279)	**70.35%**(16)	68.81%(16)	67.27%(16)

The highest value which is in bold represents the best performance of the three algorithms in each dataset. The value in parentheses are the number of selected features in each dataset. We selected the same number of features to compare by different feature selection algorithms. From table 7.2 to 7.4, we can see that FGMI-AHC has the best performance in most datasets. The performance by using all features in each dataset are shown in each table as well. Figure 7.1 shows classification accuracy comparison by different learning algorithms clearly.

Table 7.3: Classification accuracy comparison by 1-NN

NO.	All features	FGMI-AHC	IG	ReliefF
1	70.56%(9)	**71.5%**(4)	70.56%(4)	69.63%(4)
2	97.74%(16)	**94.83%**(9)	90.91%(9)	**94.83%**(9)
3	96.84%(30)	**94.8%**(10)	92.83%(10)	93.3%(10)
4	90.78%(57)	86.7%(9)	**88.76%**(9)	79.68%(9)
5	86.32%(34)	89.46%(8)	90.31%(8)	**92.02%**(8)
6	89%(36)	**86.95%**(9)	86.1%(9)	59.15%(9)
7	84.46%(41)	**83.62%**10)	79.24%10)	82.75%10)
8	77.23%(16)	**77.87%**(7)	77.45%(7)	74.68%(7)
9	66.31%(22)	**62.57%**(9)	62.03%(9)	60.43%(9)
10	61.5%(44)	**69.52%**(11)	62.57%(11)	67.38%(11)
11	97.61%(265)	**92.72%**(19)	92.59%(19)	92.47%(19)
12	85%(256)	**70.56%**(18)	68.89%(18)	64.44%(18)
13	97.94%(64)	**96.27%**(20)	94.38%(20)	96.1%(20)
14	52.88%(279)	**59.96%**(16)	58.41%(16)	56.64%(16)

Table 7.4: Classification accuracy comparison by SVM

NO.	All features	FGMI-AHC	IG	ReliefF
1	57.48%(9)	**60.75%**(4)	57.94%(4)	58.88%(4)
2	94.94%(16)	**85.71%**(6)	70.78%(6)	79.67%(6)
3	93.8%(30)	**93.88%**(10)	93.26%(10)	92.67%(10)
4	90.41%(57)	**85.66%**(9)	83.98%(9)	77.92%(9)
5	88.6%(34)	**85.19%**(8)	84.05%(8)	84.9%(8)
6	85.1%(36)	83.4%(9)	**83.5%**(9)	57.75%(9)
7	85.59%(41)	79.62%(10)	76.87%(10)	**83.22%**(10)
8	84.89%(16)	**85.11%**(7)	84.89%(7)	84.89%(7)
9	67.91%(22)	**69.52%**(9)	65.78%(9)	**69.52%**(9)
10	72.19%(44)	**70.59%**(11)	68.45%(11)	**70.59%**(11)
11	98.31%(265)	**95.1%**(19)	94.92%(19)	94.73%(19)
12	94.17%(256)	**75.09%**(18)	74.81%(18)	70.19%(18)
13	96.49%(64)	**93.32%**(20)	92.1%(20)	92.71%(20)
14	70.13%(279)	**68.58%**(16)	63.27%(16)	67.48%(16)

Table 7.5: Classification accuracy comparison by Naive Bayes

NO.	All features	FGMI-AHC	IG	ReliefF
1	57.48%(9)	**60.75%**(4)	57.94%(4)	58.88%(4)
2	82.13%(16)	72.93%(6)	60.52%(6)	**78.56%**(6)
3	92.98%(30)	**92.92%**(10)	92.81%(10)	92.56%(10)
4	79.29%(57)	**82.81%**(9)	77.9%(9)	66.36%(9)
5	82.62%(34)	88.31%(8)	89.74%(8)	**90.88%**(8)
6	79.6%(36)	78.95%(9)	**79.65%**(9)	58.45%(9)
7	75.92%(41)	**73.74%**(10)	72.8%(10)	68.15%(10)
8	78.51%(16)	**84.68%**(7)	83.83%(7)	82.77%(7)
9	60.96%(22)	**70.05%**(9)	65.78%(9)	69.52%(9)
10	69.52%(44)	**69.52%**(11)	67.91%(11)	68.45%(11)
11	92.09%(265)	**89.52%**(19)	89.39%(19)	88.39%(19)
12	93.15%(256)	**76.39%**(18)	76.02%(18)	70.09%(18)
13	89.43%(64)	**87.25%**(20)	86.59%(20)	87.15%(20)
14	52.88%(279)	**66.81%**(16)	66.37%(16)	65.71%(16)

Since FGMI-AHC algorithm is based on agglomerative hierarchical clustering, a different distance metric could produce different results. In our experiments, we use inner squared distance.

7.2 Test by OSFS

In chapter 5, we proposed two Online Streaming Feature Selection Algorithms (OSFS). OSFS1 is based on two thresholds which need to be input before using the algorithm and OSFS2 based on criteria which is calculated in the algorithm. OSFS1 requires users learn more about a dataset before using it. So we use OSFS2 to test on other datasets to compare with other algorithms.

In this test, we use same datasets as section 7.1. And we compare the performance with mRMR algorithm which is used in chapter 5. Figure 7.2, 7.3, 7.4 and 7.5 show the classification accuracy comparison between OSFS2 and mRMR by decision tree (C4.5), 1-Nearest Neighbor (1-NN), Support Vector Machines (SVM) and Naive Bayes classifiers respectively.

From the comparison we could see that OSFS2 could achieve better performance that mRMR algorithm. Table 7.6 describes comparison results between OSFS2 and mRMR. The last row of the table shows the number of selected features for each dataset.

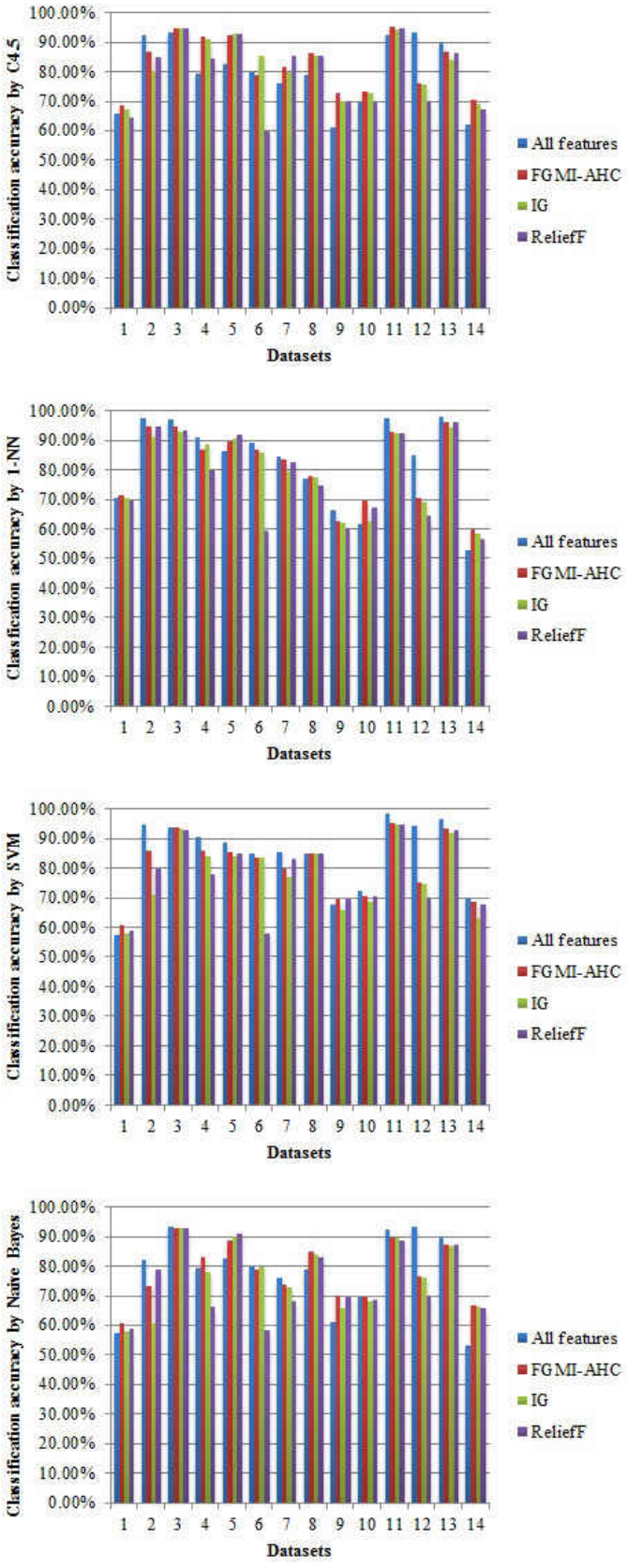

Figure 7.1: Classification accuracy comparison by different algorithms

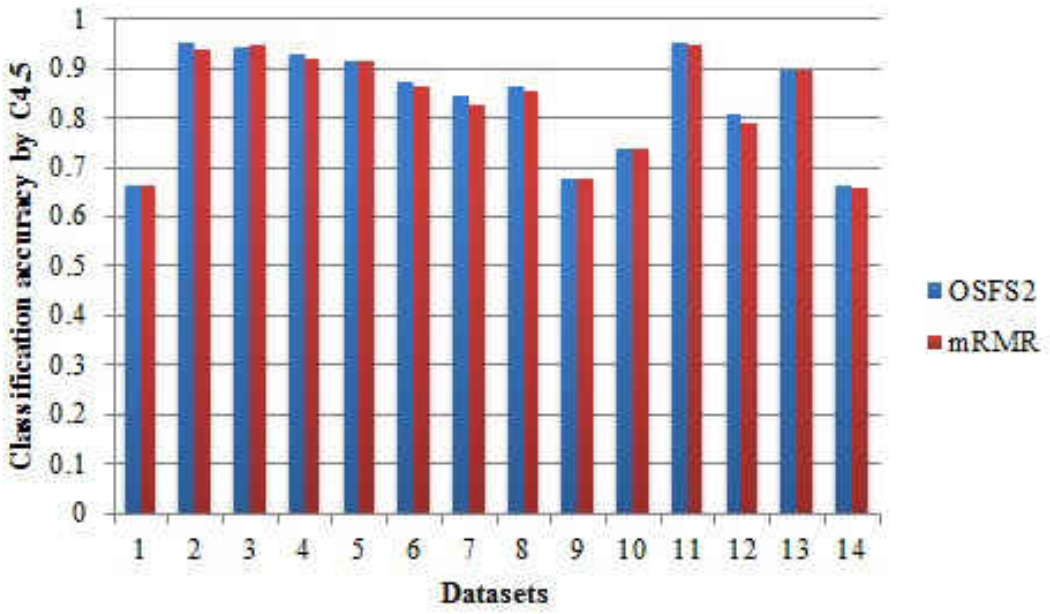

Figure 7.2: Classification accuracy between OSFS2 and mRMR by C4.5

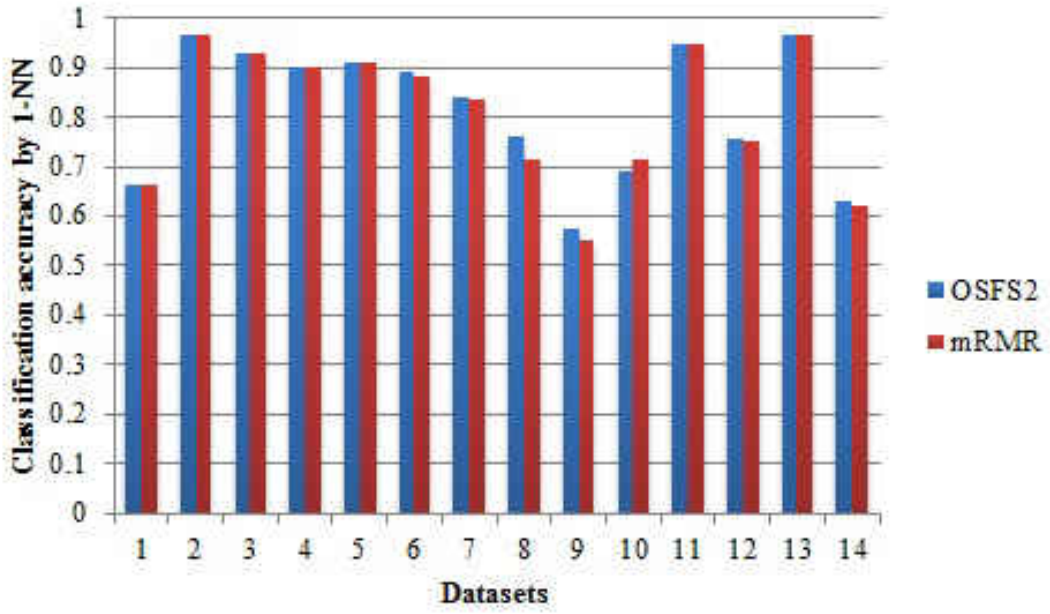

Figure 7.3: Classification accuracy between OSFS2 and mRMR by 1-NN

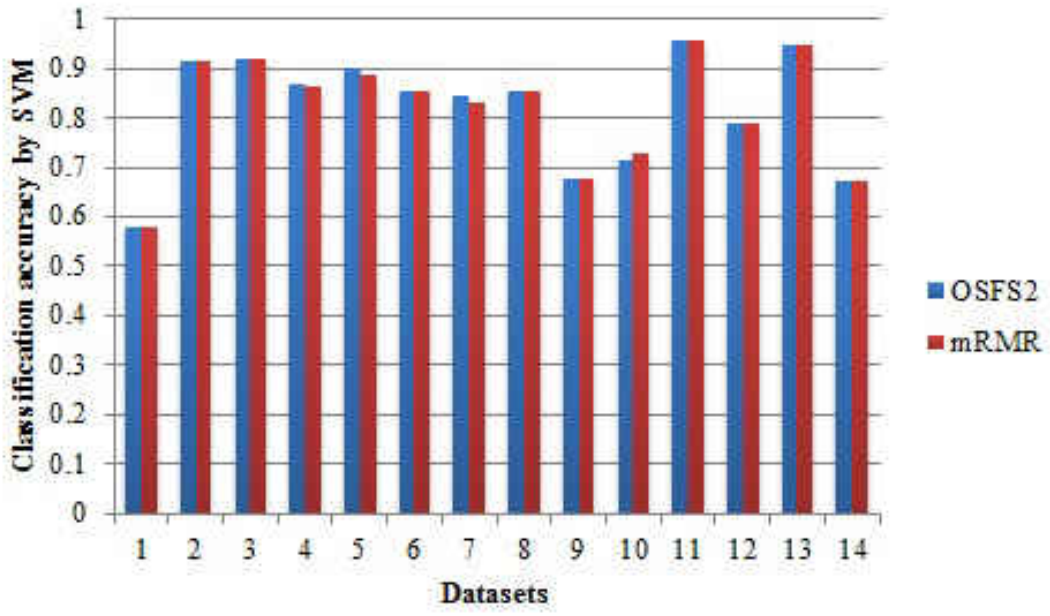

Figure 7.4: Classification accuracy between OSFS2 and mRMR by SVM

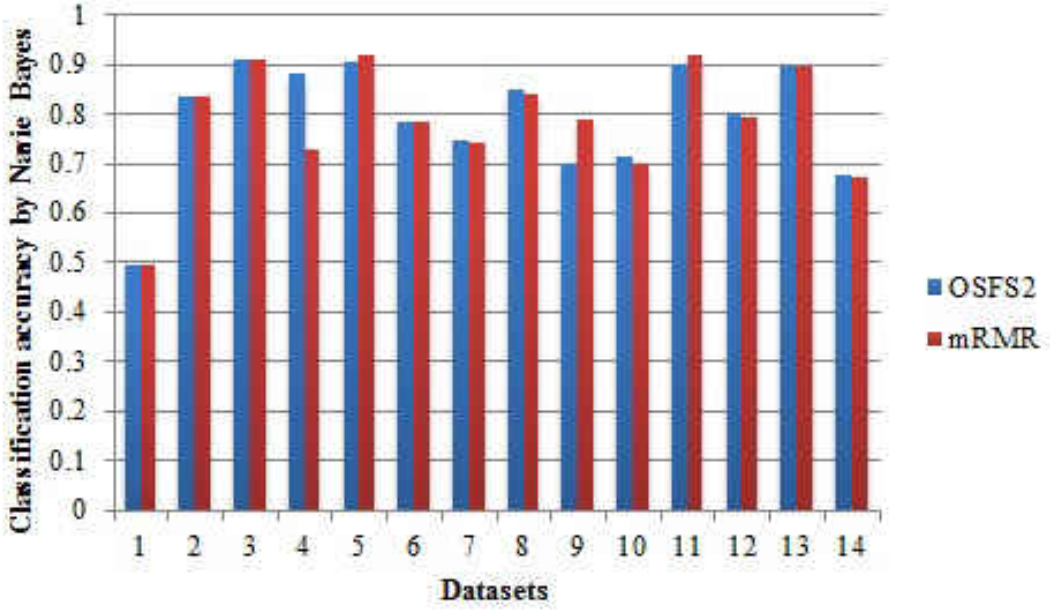

Figure 7.5: Classification accuracy between OSFS2 and mRMR by Naive Bayes

Table 7.6: Comparison Results Between OSFS2 and mRMR by different learning algorithms

Classification Algorithm	FS Algorithm	1	2	3	4	5	6	7	8	9	10	11	12	13	14
C4.5	OSFS2	66.36%	94.94%	94.25%	92.61%	91.45%	86.95%	84.17%	86.17%	67.50%	73.75%	95.10%	80.74%	89.32%	66.37%
	mRMR	66.36%	93.72%	94.60%	91.98%	91.45%	86.36%	82.27%	85.11%	67.50%	73.75%	94.54%	78.70%	89.30%	65.49%
1-NN	OSFS2	66.36%	96.48%	92.84%	89.76%	90.88%	88.76%	83.89%	75.96%	57.50%	68.75%	94.73%	75.56%	96.42%	63.05%
	mRMR	66.36%	96.40%	92.81%	89.72%	90.88%	87.98%	83.41%	71.49%	55.00%	71.25%	94.35%	75.09%	96.42%	62.17%
SVM	OSFS2	57.94%	91.27%	91.98%	86.63%	89.74%	85.14%	84.17%	85.11%	67.50%	71.25%	95.48%	78.89%	94.61%	67.04%
	mRMR	57.94%	91.22%	91.99%	86.33%	88.60%	85.10%	83.13%	85.11%	67.50%	72.50%	95.35%	78.70%	94.61%	67.04%
Naive Bayes	OSFS2	49.53%	83.49%	90.76%	87.89%	90.60%	78.44%	74.69%	84.68%	70.00%	71.25%	89.96%	80.19%	89.25%	67.48%
	mRMR	49.53%	83.28%	90.76%	72.53%	91.74%	78.11%	74.22%	83.62%	78.75%	70.00%	91.90%	79.17%	89.25%	67.26%
Number of selected features		5	8	10	15	10	12	13	7	8	11	27	26	18	28

7.3 Test by NEAC

In chapter 6, two algorithms which used unsupervised methods are proposed. New Evidence Accumulation Clustering algorithm is the more successful of the two algorithms, and so in this chapter we test New Evidence Accumulation Clustering with Hierarchical Clustering Algorithm (NEAC) on other datasets.

In this experiment, we test 8 datasets which are from section 7.1. They are showed in table 7.7. All of the datasets in table 7.7 have 2 class. The reason is that NEAC has a limitation that it could only solve 2-class problem.

Table 7.7: Datasets for test in experiments

NO.	Dataset name	Instance NO.	Feature NO.
1	Phishing Websites	11055	30
2	Spambase	4601	57
3	Ionosphere	351	34
4	Biodeg	1055	41
5	ThoracricSurgery	470	16
6	SPECT	187	22
7	SPECTF	187	44
8	Semeion	1593	265

We compare proposed NEAC algorithm with Ada's algorithm which is used in Chapter 6 to compare performance evaluation. Figure 7.6 shows accuracy comparison between NEAC and Ada's algorithm. From the comparison, we can see that accuracy is very close between the two algorithms in each dataset. The detailed accuracy values are illustrated in table 7.8.

Table 7.8: Accuracy comparison between NEAC and Ada's algorithm

NO.	Dataset name	Ada's algorithm	NEAC
1	Phishing Websites	55.71%	55.89%
2	Spambase	60.57%	60.57%
3	Ionosphere	63.82%	64.39%
4	Biodeg	66.16%	66.16%
5	ThoracricSurgery	85.11%	84.89%
6	SPECT	54.55%	55.61%
7	SPECTF	92.51%	92.51%
8	Semeion	90.02%	90.4%

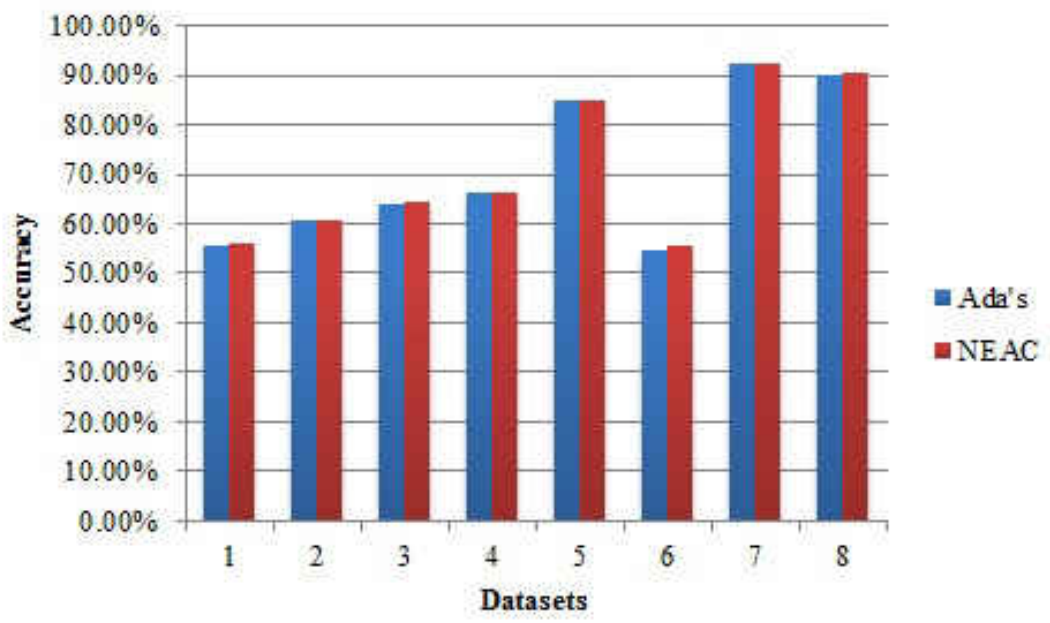

Figure 7.6: Accuracy comparison between NEAC and Ada's algorithm

In this test, we use single link method over the similarity matrix for producing dendrogram in both of the two algorithms. And we set the number of clusters in K-means algorithm to 60. And in Ada's algorithm, we run 60 times to get the co-association matrix in every dataset since the feature number of most datasets in table 7.7 are less than 60. From the accuracy comparison in table 7.7, we can see that the values are very close among the eight group data. Three of them are equal, NEAC won four of them and lost one.

To investigate the impact of the parameters and settings in NEAC. The distance between clusters in hierachical clustering we used is single in above tests. We use another two different distance complete and average to compare. Single, complete and average are shortest, furthest and unweighted average distance between clusters respectively. And accuracy comparison by NEAC on different parameters are shown in figure 7.9. The last column shows different k values in NEAC algorithm, we change $k = 2$ to $k = 60$ and the results are shown in figure 7.9. And from the results we can see that k value has litte impact on the result. And different distance between clusters in hierachical clustering impacted the results very much.

Table 7.9: Accuracy comparison by NEAC on different parameters

NO.	Dataset name	NEAC			
		Single	Complete	Average	k=60
1	Phishing Websites	55.89%	54.08%	52.71%	55.89%
2	Spambase	60.57%	83.46%	75.64%	60.57%
3	Ionosphere	64.39%	63.82%	62.96%	64.39%
4	Biodeg	66.16%	75.83%	63.41%	66.16%
5	ThoracicSurgery	84.89%	57.02%	50.21%	84.47%
6	SPECT	55.61%	58.82%	58.82%	54.55%
7	SPECTF	92.51%	95.19%	92.51%	92.51%
8	Semeion	90.4%	58.44%	57.69%	90.4%

Chapter 8

Conclusions

In this chapter, a high level summary of the research as detailed in the preceding chapters will be presented. The thesis has demonstrated some feature selection algorithms on KDD 99 dataset after reviewed relevant approaches in the literature. After that, we compared the results with either the original approaches or relevant techniques in the literature. The most promising of the proposed algorithms has also been tested on other datasets. This chapter also presents a number of initial thoughts about the directions for future research.

8.1 Summary of Thesis

Detailed descriptions and statistical observations of KDD 99 dataset are introduced in chapter 1. Since the dataset is widely used dataset for intrusion detection, it is used to test proposed feature selection algorithms. Datasets of intrusion detection area usually have large instances and features. Take KDD 99 dataset for example, it has 5 millions instances in training dataset and we could only use 10 % of it to test our proposed algorithms.

A survey of feature selection and intrusion detection system has been given in chapter 2. A brief history and classification of IDS are presented. After that, it states some approaches and challenges to intrusion detection. Feature selection part is very important in this thesis. In chapter 2, feature selection process and four evaluation measures are described. And four feature selection approaches are also presented, they are used to compare with my proposed algorithms in other chapters.

Modified mutual information-based feature selection algorithms has been proposed in chapter 3. Mutual information is specifically introduced in the chapter since this algorithm is based on mutual information (MI) theory. The mutual information between each feature and class label shown the relationship between features and class label. MMFS algorithm considered MI not only between each feature and class label, but also between two features. At last, MMFS was compared with DMIFS which is proposed by Huawen in 2009. Computation comparison are given as well in this chapter.

Chapter 4 presented two feature grouping algorithms based on mutual information. FGMI uses fuzzy C means clustering algorithm to get groups. And FGMI-AHC algorithm creates groups using agglomerative hierarchical clustering method. The two algorithms are compared with DMIFS and MMFS by different number of selected features. And they are also compared by different classification algorithms. And from the comparison, we can see that FGMI-AHC could achieve better performance than FGMI.

Chapter 5 states two online streaming feature selection algorithms, OSFS1 and OSFS2. They are based on the analysis of feature relevance and redundancy. OSFS1 uses threshold to decide whether a feature is redundant or relevant with labels and other features. And OSFS1 limits its application as users need to understand the candidate dataset before using it. OSFS2 utilizes a criterion which is computed by relevance and redundancy values. And it has more universal applicability.

Two algorithms in chapter 6 use unsupervised method on kdd 99 dataset. Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm combines fuzzy C means and C4.5 together. And it divides data into two parts according to menbership which is got from clustering result by fuzzy C means. Then, it classifies data in each part separately. NEAC algorithm is based on evidence accumulation clustering, and it clusters each column of a dataset, rather than a whole dataset.

In chapter 7, we test our proposed algorithms on other datasets. We got 14 datasets from UCI machine learning repository. And we test three algorithms, FGMI-AHC, OSFS2 and NEAC. From the comparison results, we could see that our proposed algorithms are effective and could get better performance.

For all the algorithms we proposed, OSFS2 could achieve the best performance on KDD 99 dataset. And cascading Fuzzy C Means Clustering and C4.5 Decision

Tree Classification Algorithm get the worst performance. And the results of MMIFS, FGMI, FGMI-AHC, OSFS and Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification Algorithm come to the expected target, but NEAC does not. The reason is that consensus function we used in NEAC is k-means clustering method and result in the performance are not significant compare to Ada's algorithm.

I think future directions in this area should include the following aspects. Firstly, feature selection is a kind of dimensionality reduction. And dimensionality reduction is not only for features but also for instance. Like feature selection, instance should be selected and it might be used in IDS area. Moreover, traditional feature selection algorithms are not applicable for big data and variable dimension datasets whose feature and instance numbers are not fixed. OSFS is a method could solve this problem and more similar methods are needed. At last, the relationships of features is very important in this area, it determines feature selection strategies. Thus, designing more effective measurements to measure the relationship between features is vital direction as well.

8.2 Future Works

From chapter 3 to 6, we proposed some feature selection algorithms and unsupervised methods for intrusion detection area. In chapter 7, we could see that these algorithm could be applied on other datasets and other areas. Some of them need to be improved and some of them need to be expanded and applied in other areas. In this section, future works are discussed.

8.2.1 Short Term Tasks

The algorithms proposed in the thesis are good enough and some of them need to improved or expanded. This section will propose some potential work for some algorithms.

8.2.1.1 MMFS

Most proposed algorithms use mutual information to measure the relationship between features in this thesis. MMFS algorithm is the most typical one. We could find other metrics or methods to measure instead of mutual information and compare the

results with different measurements. And methods which could measure relationship between features could be used, such as correlation. Some of measurements could be used to produce a similarity matrix and create groups for feature grouping algorithms.

8.2.1.2 FGMI

The selecting strategy of feature grouping used in this algorithm is selecting one feature from each group. As it is discussed in chapter 4, we could design different selecting strategies for feature grouping. Figure 8.1 shows outline of improved selecting strategy of feature grouping. In improved scheme, we can select different number of features from one group based on some selecting strategies. For example, we could select 2 or more features from a group and the number of selected features could be different from a group to other groups. And selecting should be help for classification rather than random selecting. As stated above, different measurements also could be used to create groups and it could combine with selecting strategies.

8.2.1.3 OSFS

The criterion of online streaming feature selection algorithm could be considered to revise by more reasonable methods. Current criterion of OSFS2 is based on relevance and mutual information between features. If we use other methods to change the criterion, the results of OSFS2 will be changed. The criterion we used now is related to the measurement between features. And criteria should reflect whether a streaming feature could help the selected feature for classification. Thus, the relationship between a streaming feature and other features and class labels are very important to construct the criterion. As we discussed in chapter 5, the results of OSFS will vary depending on the order in which features are streamed. Therefore, the criterion should to reduce the influence of it.

8.2.2 Long Term Developments

In this section, some ideas about IDS and related algorithm improvements are discussed. And this work could be seen as long term developments.

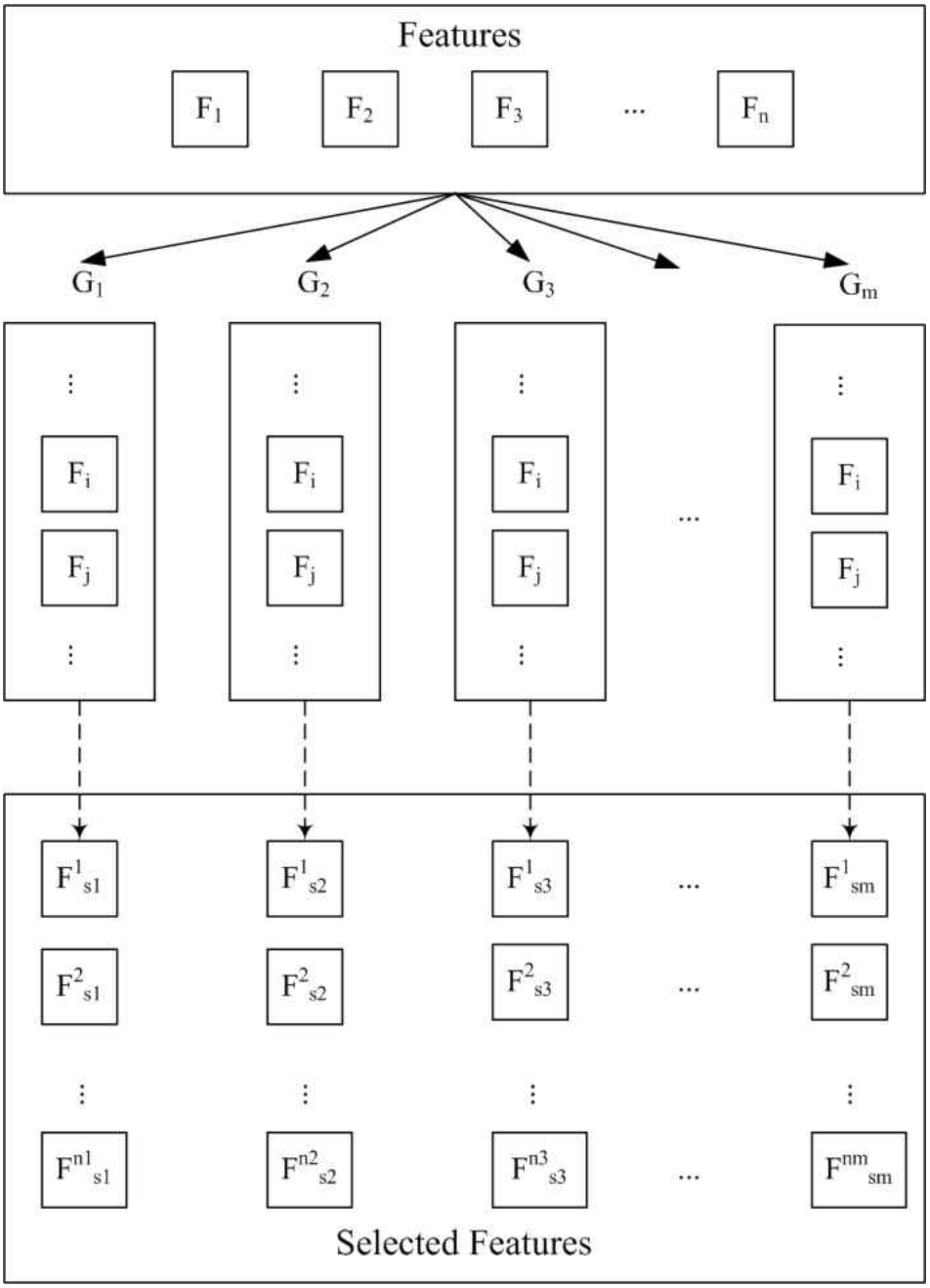

Figure 8.1: Improved selecting strategy of feature grouping

8.2.2.1 Unsupervised methods for IDS

In chapter 6, we proposed two algorithms related to unsupervised learning method to deal with IDS. Cascading Fuzzy C Means Clustering and C4.5 Decision Tree Classification algorithm could only used for two class labeled dataset and it needs to be improved in the future. One advantage of NEAC algorithm is it clustering features one by one. But how to store the matrix when data growing is a problem to be solved in future.

8.2.2.2 Application to Big Data

Both OSFS and NEAC algorithms deal with dataset by features. OSFS streaming a feature every time and NEAC cluster one feature at a time. This characteristic could be used on big data area which is a hot area recently. But it still needs to solve many problems, such as data storage, intermediate results storage and processing and so on.

8.2.2.3 Real-time IDS

All the proposed algorithms and methods in this thesis could only be used for static dataset of IDS. None of them could applied on real-time intrusion detection system. Real-time IDS requres response in time and it usually deals with data according to instance. Machine learning methods could be used on real-time IDS, but it needs more improvements and testing in practical.

Appendix A

Publications Arising from the Thesis

A number of publications have been generated from the research carried out within the PhD project. Below lists the resultant publications that are in close relevance to the thesis.

1. Jingping Song, Zhiliang Zhu, Peter Scully and Chris Price, "Selecting Features for Anomaly Intrusion Detection A Novel Method using Fuzzy C Means and Decision Tree Classification", Cyberspace Safety and Security. Springer International Publishing, 2013: 299-307.
2. Jingping Song, Zhiliang Zhu, Peter Scully and Chris Price, "Modified Mutual Information-based Feature Selection for Intrusion Detection Systems in Decision Tree Learning", Journal of computers, 2014, 9(7): 1542-1546.
3. Jingping Song, Zhiliang Zhu and Chris Price, Feature Grouping for Intrusion Detection System Based on Hierarchical Clustering, Availability, Reliability, and Security in Information Systems, Springer International Publishing, 2014: 270-280.
4. Jingping Song, Zhiliang Zhu and Chris Price, "Feature Grouping for Intrusion Detection Based on Mutual Information," Journal of Communications, vol. 9, no. 12, pp. 987-993, 2014.
5. Scully P, Song J, Disso J P, et al. "CARDINAL-E: AIS Extensions to CARDINAL for Decentralised Self-Organisation for Network Security", Advances in Artificial Life, ECAL. 2013, 12: 1235-1236.

Appendix B

List of Acronyms

1-NN 1-Nearest Neighbor

CFS Correlation based Feature Selection

DMIFS Feature Selection using Dynamic Mutual Information

DR Detection Rate

FCM Fuzzy C Means

FGMI Feature Grouping for Intrusion Detection based on Mutual Information

FGMI-AHC Feature Grouping by Agglomerative Hierarchical Clustering based on Mutual Information

FN False Negative

FP False Positive

FPR False Positive Rate

FS Feature Selection

HIDS Host based Intrusion Detection System

IDS Intrusion Detection System

IG Information Gain

MMFS Modified Mutual Information-based Feature Selection

mRMR Maximum Relevance Minimum Redundancy

NEAC New Evidence Accumulation Clustering Algorithm

NIDS Network based Intrusion Detection System

P Precision

R Recall

OSFS Online Streaming Feature Selection

OSFS1 Online Streaming Feature Selection Algorithm 1

OSFS2 Online Streaming Feature Selection Algorithm 2

SBE Sequential Backward Elimination

SFS Sequential Forward Selection

SVM Support Vector Machines

TA Total Accuracy

TN True Negative

TP True Positive

TPR True Positive Rate

Bibliography

[1] M. S. Abadeh, J. Habibi, and C. Lucas, "Intrusion detection using a fuzzy genetics-based learning algorithm," *Journal of Network & Computer Applications*, vol. 30, no. 1, pp. 414–428, 2007.

[2] H. Altwaijry, "Bayesian based intrusion detection system," in *IAENG Transactions on Engineering Technologies*. Springer, 2013, pp. 29–44.

[3] F. Amiri, Y. Rezaei, M. Mohammad, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011.

[4] J. P. Anderson, "Computer security threat monitoring and surveillance," *Fort Washington*, 1980.

[5] R. J. Anderson, "Security engeering: A guide to building dependable distributed systems," *Wiley & Sons*, vol. 2, no. 1, pp. 115–117, 2001.

[6] R. Anderson and A. Khattak, "The use of information retrieval techniques for intrusion detection," *Proceedings of Raid*, 1998.

[7] F. Anjum, D. Subhadrabandhu, and S. Sarkar, "Signature based intrusion detection for wireless ad-hoc networks: A comparative study of various routing protocols," in *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, vol. 3. IEEE, 2003, pp. 2152–2156.

[8] B. A. Asuncion and D. J. Newman, "Uci machine learning repository, 2007. bojan cestnik. estimating probabilities: A crucial task," in *in Machine Learning*, 2012.

[9] W. H. Au, K. C. C. Chan, A. K. C. Wong, and Y. Wang, "Attribute clustering for grouping, selection, and classification of gene expression," in *IEEE/ACM Transactions on Computational Biology and Bioinformations*, 2005, pp. 83–101.

[10] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 3, pp. 186–205, 2000.

[11] Y. Bai and H. Kobayashi, "Intrusion detection systems: technology and development," in *Advanced Information Networking and Applications, 2003. AINA 2003. 17th International Conference on*, 2003, pp. 710–715.

[12] I. Balepin, S. Maltsev, J. Rowe, and K. Levitt, "Using specification-based intrusion detection for automated response," in *Recent Advances in Intrusion Detection*. Springer, 2003, pp. 136–154.

[13] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 537–550, 1994.

[14] D. S. Bauer and M. E. Koblentz, "Nidx- an expert system for real-time network intrusion detection," *Proceedings of the Computer Networking Symposium*, pp. 98 – 106, 1988.

[15] R. Beghdad, "Modelling intrusion detection as an allocation problem," *Pattern Recognition Letters*, vol. 30, no. 8, pp. 774–779, 2009.

[16] M. Bernaschi, E. D. Aiutolo, and P. Rughetti, "Enforcing network security: a real cease study in a research organization," *Computers & Security*, vol. 18, pp. 533–543, 1999.

[17] A. Bivens, R. Smith, M. Embrechts, C. Palagiri, and B. Szymanski, "Network-based intrusion detection using neural networks," *Proc. ANNIE 2002 Conference*, pp. 10–13, 2002.

[18] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "Feature selection and classification in multiple class datasets: An application to kdd cup 99 dataset," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5947–5957, 2011.

[19] S. Boutemedjet, N. Bouguila, and D. Ziou, "A hybrid feature extraction selection approach for high-dimensional non-gaussian data clustering," *Pattern Analysis & Machine Intelligence IEEE Transactions on*, vol. 31, no. 8, pp. 1429–1443, 2008.

[20] Z. Y. Cai, Y. Gan, Y. P. Jiang, and W. J. Zhong, "Development of intelligent intrusion detection based on biosimulation," *Journal of Zhengzhou University of Light Industry*, 2010.

[21] P. Casas, J. Mazel, and P. Owezarski, *UNADA: Unsupervised Network Anomaly Detection Using Sub-space Outliers Ranking*. Springer Berlin Heidelberg, 2011.

[22] ——, "Unsupervised network intrusion detection systems: Detecting the unknown without knowledge," *Computer Communications*, vol. 35, no. 7, pp. 772–783, 2012.

[23] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers & Security*, vol. 24, no. 4, pp. 295–307, 2005.

[24] C. M. Chen, Y. L. Chen, and H. C. Lin, "An efficient network intrusion detection," *Computer Communications*, vol. 33, no. 4, pp. 477–484, 2010.

[25] T. M. Chen, J. Blasco, J. Alzubi, and O. Alzubi, "Intrusion detection," *Engineering & Technology Reference*, vol. 1, 2014.

[26] W. Chen, "Design of a worm isolation and unknown worm monitoring system based on honeypot," 2014.

[27] W. Chimphlee, A. H. Abdullah, and M. N. Md Sap, "Unsupervised anomaly detection with unlabeled data using clustering," *Computer Security Anomaly Detection Unsupervised Clustering Outliers Unlabeled Data*, 2005.

[28] J. Cho, C. Lee, S. Cho, J. H. Song, J. Lim, and J. Moon, "A statistical model for network data analysis: Kdd cup 99 data evaluation and its comparing with mit lincoln laboratory network data," *Simulation Modelling Practice and Theory*, vol. 18, no. 4, pp. 431–435, 2010.

[29] T. W. S. Chow and . Huang, D., "Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information." *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 213–24, 2005.

[30] K. Chrysostomou, S. Y. Chen, and X. Liu, "Combining multiple classifiers for wrapper feature selection," *International Journal of Data Mining Modelling & Management*, no. 1, pp. 91–102, 2009.

[31] R. L. Cilibrasi and P. M. B. Vitanyi, "A fast quartet tree heuristic for hierarchical clustering," *Pattern Recognition*, vol. 44, no. 3, pp. 662–677, 2014.

[32] G. Cormode and S. Muthukrishnan, "What's new: Finding significant differences in network data streams," in *in Proc. of IEEE Infocom*, 2004, pp. 1534–1545.

[33] H. P. Corporation, "A profile based network intrusion detection and prevention system for securing cloud environment," *International Journal of Distributed Sensor Networks*, vol. 2013, no. 1, pp. 8–10, 2013.

[34] M. Crosbie and E. H. Spafford, "Defending a computer system using autonomous agents," in *IN PROCEEDINGS OF THE 18TH NATIONAL INFORMATION SYSTEMS SECURITY CONFERENCE*, 1996.

[35] D. Dasgupta, "Immunity-based intrusion detection system: a general framework," in *Proc. of the 22nd NISSC*, vol. 1, 1999, pp. 147–160.

[36] M. Dash and H. Liu, "Consistency-based search in feature selection," *Artificial Intelligence*, vol. 151, no. 22, pp. 155–176, 2003.

[37] J. J. Davis and A. J. Clark, "Data preprocessing for anomaly based network intrusion detection: A review," *Computers & Security*, vol. 30, pp. 353–375, 2011.

[38] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer Networks*, vol. 31, no. 8, pp. 805–822, 1999.

[39] D. E. Denning and P. G. Neumann, "Requirements and model for ides a real-time intrusion-detection expert system," in *Document A005, SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025*, 1985.

[40] D. E. Denning, "An intrusion-detection model," *Software Engineering, IEEE Transactions on*, no. 2, pp. 222–232, 1987.

[41] O. Devos and L. Duponchel, "Parallel genetic algorithm co-optimization of spectral pre-processing and wavelength selection for pls regression," *Chemometrics & Intelligent Laboratory Systems*, vol. 107, no. 1, pp. 50–58, 2011.

[42] S. Dharmapurikar, J. Lockwood, and M. Ieee, "Fast and scalable pattern matching for network intrusion detection systems," in *IEEE Journal on Selected Areas in Communications*, 2006.

[43] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, *A Geometric Framework for Unsupervised Anomaly Detection*. Springer US, 2002.

[44] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *KDD*, vol. 96, 1996, pp. 226–231.

[45] P. A. Estevez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *Neural Networks IEEE Transactions on*, vol. 20, no. 2, pp. 189–201, 2009.

[46] B. Fish, A. Khan, N. H. Chehade, C. Chien, and G. Pottie, "Feature selection based on mutual information for human activity recognition," *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, vol. 37, no. 4, pp. 1729–1732, 2012.

[47] A. Fred and A. K. Jain, "Evidence accumulation clustering based on the k-means algorithm," in *Structural, syntactic, and statistical pattern recognition*. Springer, 2002, pp. 442–451.

[48] A. L. Fred and A. K. Jain, "Combining multiple clusterings using evidence accumulation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 6, pp. 835–850, 2005.

[49] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, pp. 18–28, 2009.

[50] J. Georges and A. H. Milley, "Kdd'99 competition: Knowledge discovery contest," *ACM SIGKDD Explorations Newsletter*, vol. 1, no. 2, pp. 79–84, 2000.

[51] G. Giacinto, R. Perdisci, M. Del Rio, and F. Roli, "Intrusion detection in computer networks by a modular ensemble of one-class classifiers," *Information Fusion Special Issue on Applications of Ensemble Methods*, vol. 9, no. 1, pp. 69–82, 2008.

[52] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern recognition letters*, vol. 24, no. 12, pp. 1795–1803, 2003.

[53] W. Gong, X. H. Zeng, and L. I. Liu-Bai, "Application of immune principle in adaptive intrusion detection," *Computer Engineering & Design*, 2008.

[54] V. Gowadia, C. Farkas, and M. Valtorta, "Paid: A probabilistic agent-based intrusion detection system," in *IN COMPUTERS & SECURITY*, 2005, pp. 529–545.

[55] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, "Feature extraction, foundations and applications," *Studies in Fuzziness & Soft Computing*, 2006.

[56] W. U. Hai-Bo, J. S. Gao, Q. T. Tang, and Y. Zhang, "Research on network intrusion detection system based on biological immune principle," *Computer Technology & Development*, 2013.

[57] M. A. Hall and L. A. Smith, "Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper," in *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, 1999, pp. 235–239.

[58] S. Hansman and R. Hunt, "A taxonomy of network and computer attacks," *Computers & Security*, vol. 24, pp. 31–43, 2005.

[59] S. A. Hofmeyr, S. Forrest, and A. Somayaji, "Intrusion detection using sequences of system calls," *Journal of computer security*, vol. 6, no. 3, pp. 151–180, 1998.

[60] H. Holm, "Performance of automated network vulnerability scanning at remediating security issues," *Computers & Security*, vol. 31, no. 2, pp. 164–175, 2012.

[61] S. J. Horng, M. Y. Su, Y. H. Chen, T. W. Kao, R. J. Chen, J. L. Lai, and C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *Expert Systems with Applications*, vol. 38, no. 1, pp. 306–313, 2011.

[62] Y. Hou and X. F. Zheng, "Expert system based intrusion detection system," *International Conference on Information Management Innovation Management & Industrial Engineering*, vol. 4, pp. 404–407, 2010.

[63] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "Hybrid feature selection by combining filters and wrappers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8144–8150, 2011.

[64] Q. Hu, X. Che, L. Zhang, and D. Yu, "Feature evaluation and selection based on neighborhood soft margin," *Neurocomputing*, vol. 73, no. s 10âĂŞ12, pp. 2114–2124, 2010.

[65] W. Hu, J. Gao, Y. Wang, O. Wu, and S. Maybank, "Online adaboost-based parameterized methods for dynamic distributed network intrusion detection," *Cybernetics, IEEE Transactions on*, vol. 44, no. 1, pp. 66–82, 2014.

[66] W. Hua-Liang and S. A. Billings, "Feature subset selection and ranking for data dimensionality reduction." *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 29, no. 1, pp. 162–166, 2007.

[67] J. Huang, Y. Cai, and X. Xu, "A hybrid genetic algorithm for feature selection wrapper based on mutual information," *Pattern Recognition Letters*, vol. 28, no. 13, pp. 1825–1844, 2007.

[68] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection approach," *Software Engineering IEEE Transactions on*, vol. 21, no. 3, pp. 181–199, 1995.

[69] I. Inza, P. LarraÃśaga, R. Etxeberria, and B. Sierra, "Feature subset selection by bayesian network-based optimization," *Artificial Intelligence*, vol. 123, no. s 1âĂŞ2, pp. 157–184, 2000.

[70] K. A. Jackson, D. H. Dubois, and C. A. Stallings, "An expert system application for network intrusion detection," *Proceedings of National Computer Security Conference*, pp. 234–237, 1990.

[71] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 2, pp. 153–158, 1997.

[72] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[73] R. Jensen, "Combining rough and fuzzy sets for feature selection," Ph.D. dissertation, University of Edinburgh, 2005.

[74] Y. S. Jeong, I. H. Kang, M. K. Jeong, and D. Kong, "A new feature selection method for one-class classification problems," *IEEE Transactions on Systems Man & Cybernetics Part C*, vol. 42, no. 6, pp. 1500–1509, 2012.

[75] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant feature and the subset selection problem," in *Rutgers University*, 1994.

[76] K. Juszczyszyn, N. T. Nguyen, G. Kolaczek, A. Grzech, and A. Katarzyniak, "Agent-based approach for distributed intrusion detection system design." *Lecture Notes in Computer Science*, pp. 224–231, 2006.

[77] V. Jyothsna, V. R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems," *International Journal of Computer Applications*, vol. 28, no. 7, pp. 26–35, 2011.

[78] M. E. Kabir and J. Hu, "A statistical framework for intrusion detection system," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2014 11th International Conference on*, 2014, pp. 941–946.

[79] M. M. Kabir, M. M. Islam, and K. Murase, "A new wrapper feature selection approach using neural network," *Neurocomputing*, vol. 73, no. s 16âĂŞ18, pp. 3273–3283, 2010.

[80] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, "Selecting features for intrusion detection: a feature relevance analysis on kdd 99 intrusion detection datasets," in *Proceedings of the third annual conference on privacy, security and trust*. Citeseer, 2005.

[81] R. A. Kemmerer and G. Vigna, "Intrusion detection: a brief history and overview," *Computer*, vol. 35, no. 4, pp. 27 – 30, 2002.

[82] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[83] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proceedings of the ninth international workshop on Machine learning*, 1992, pp. 249–256.

[84] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial intelligence*, vol. 97, no. 1, pp. 273–324, 1997.

[85] I. Kojadinovic, "Agglomerative hierarchical clustering of continuous variables based on mutual information," *Computational Statistics & Data Analysis*, vol. 46, no. 2, pp. 269–294, 2004.

[86] C. Kolias, G. Kambourakis, and M. Maragoudakis, "Swarm intelligence in intrusion detection: A survey," *Computers & Security*, vol. 30, no. 8, pp. 625–642, 2011.

[87] I. Kononenko and I. Kononenko, "Analysis and extension of relief," in *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 1994.

[88] I. Kononenko, "Estimating attributes: analysis and extensions of relief," in *Machine Learning: ECML-94*. Springer, 1994, pp. 171–182.

[89] I. Kononenko, E. Šimec, and M. Robnik-Šikonja, "Overcoming the myopia of inductive learning algorithms with relieff," *Applied Intelligence*, vol. 7, no. 1, pp. 39–55, 1997.

[90] C. Kruegel and T. Toth, "Using decision trees to improve signature-based intrusion detection," in *Recent Advances in Intrusion Detection*. Springer, 2003, pp. 173–191.

[91] C. Krügel, T. Toth, and E. Kirda, "Service specific anomaly detection for network intrusion detection," in *Proceedings of the 2002 ACM symposium on Applied computing*. ACM, 2002, pp. 201–208.

[92] A. Kumarshrivas and A. Kumar Dewangan, "An ensemble model for classification of attacks with feature selection based on kdd99 and nsl-kdd data set," *International Journal of Computer Applications*, vol. 99, no. 15, pp. 8–13, 2014.

[93] N. Kwak and C. H. Choi, "Input feature selection by mutual information based on parzen window," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 24, no. 12, pp. 1667 – 1671, 2002.

[94] L. Lan and S. Vucetic, "A multi-task feature selection filter for microarray classification," in *2009 IEEE International Conference on Bioinformatics and Biomedicine*, 2009, pp. 160–165.

[95] T. Lane and C. E. Brodley, "An application of machine learning to anomaly detection," in *In Proceedings of the 20th National Information Systems Security Conference*, 1997, pp. 366–380.

[96] T. D. Lane, "Machine learning techniques for the computer security domain of anomaly detection," "*Machine learning techniques for the computer security domain of anomal*" *by Terran D Lane*, 2001.

[97] P. Langley, "Selection of relevant features in machine learning," *Proceedings of the Aaai Fall Symposium on Relevance*, pp. 140–144, 1994.

[98] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection." in *SDM*. SIAM, 2003, pp. 25–36.

[99] C. Lee and G. G. Lee, "Information gain and divergence-based feature selection for machine learning-based text categorization," *Information Processing & Management*, vol. 42, no. 1, pp. 155–165, 2006.

[100] J. H. Lee, J. H. Lee, S. G. Sohn, J. H. Ryu, and T. M. Chung, "Effective value of decision tree with kdd 99 intrusion detection datasets for intrusion detection system," in *Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on*, 2008, pp. 1170–1175.

[101] M. C. Lee, "Using support vector machine with a hybrid feature selection method to the stock trend prediction," *Expert Systems with Applications*, vol. 36, no. 8, pp. 10 896–10 904, 2009.

[102] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM transactions on Information and system security (TiSSEC)*, vol. 3, no. 4, pp. 227–261, 2000.

[103] W. Lee, S. J. Stolfo, and K. W. Mok, "A data mining framework for building intrusion detection models," in *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*. IEEE, 1999, pp. 120–132.

[104] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*. Australian Computer Society, Inc., 2005, pp. 333–342.

[105] I. Levin, "Kdd-99 classifier learning contest: Llsoft's results overview," *SIGKDD explorations*, vol. 1, no. 2, pp. 67–75, 2000.

[106] J. Liang, K. S. Chin, and C. D. R. C. M. Yam, "A new method for measuring uncertainty and fuzziness in rough set theory," *International Journal of General Systems*, vol. 31, no. 4, pp. 331–342, 2002.

[107] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[108] S. W. Lin, K. C. Ying, S. C. Chen, and Z. J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines." *Expert Systems with Applications*, vol. 35, no. 4, pp. 1817–1824, 2008.

[109] S.-W. Lin, K.-C. Ying, C.-Y. Lee, and Z.-J. Lee, "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection," *Applied Soft Computing*, vol. 12, no. 10, pp. 3285–3290, 2012.

[110] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 darpa off-line intrusion detection evaluation," *Computer networks*, vol. 34, no. 4, pp. 579–595, 2000.

[111] R. P. Lippmann and R. K. Cunningham, "Improving intrusion detection performance using keyword selection and neural networks," *Computer Networks*, vol. 34, no. 4, pp. 597–603, 2000.

[112] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, vol. 2. IEEE, 2000, pp. 12–26.

[113] H. Liu and H. Motoda, *Feature selection for knowledge discovery and data mining*. Springer Science & Business Media, 2012, vol. 454.

[114] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 17, no. 4, pp. 491–502, 2005.

[115] H. Liu, J. Sun, L. Liu, and H. Zhang, "Feature selection with dynamic mutual information," *Pattern Recognition*, vol. 42, no. 7, pp. 1330–1339, 2009.

[116] X. Liu, B. Lang, Y. Xu, and B. Cheng, "Feature grouping and local soft match for mobile visual search," *Pattern Recognition Letters*, vol. 33, no. 3, pp. 239–246, 2012.

[117] Z. Y. Liu and J. Wang, "A research into an intrusion detection system based on immune principle and multi-agent in wsn," *Advanced Materials Research*, vol. 433-440, pp. 5157–5161, 2012.

[118] T. F. Lunt, "A survey of intrusion detection techniques," *Computers & Security*, vol. 12, no. 4, pp. 405–418, 1993.

[119] T. F. Lunt, R. Jagannathan, R. Lee, S. Listgarten, D. L. Edwards, P. G. Neumann, H. S. Javitz, and A. Valdes, "Ides: The enhanced prototype, a real-time intrusion detection system," *SRI Project 4185-010, SRI-CSL-88-12, CSL SRI International, Computer Science Laboratory, SRI Intl. 333 Ravenswood Ave., Menlo Park, CA 94925-3493*, 1988.

[120] M. Mahboubian and N. I. Udzir, "A naturally inspired statistical intrusion detection model," *International Journal of Computer Theory & Engineering*, 2013.

[121] S. Maldonado and R. Weber, "A wrapper method for feature selection using support vector machines," *Information Sciences*, vol. 179, no. 13, pp. 2208–2217, 2009.

[122] ——, "Embedded feature selection for support vector machines: State-of-the-art and future challenges," *Lecture Notes in Computer Science*, pp. 304–311, 2011.

[123] J. MartÃ◌nez Sotoca and F. Pla, "Supervised feature selection by clustering using conditional mutual information-based distances," *Pattern Recognition*, vol. 43, no. 6, pp. 2068–2081, 2010.

[124] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM transactions on Information and system Security*, vol. 3, no. 4, pp. 262–294, 2000.

[125] J. McHugh, A. Christie, and J. Allen, "Defending yourself: The role of intrusion detection systems," *IEEE software*, no. 5, pp. 42–51, 2000.

[126] G. Meera and S.K.Srivatsa, "Detecting and preventing attacks using network intrusion detection systems," *International Journal of Computer Science and Security*, vol. 2, no. 1, pp. 50–60, 2008.

[127] M. Mehdi, S. Zair, A. Anou, and M. Bensebti, "A bayesian networks in intrusion detection systems," *Journal of Computer Science*, vol. 2, no. May, pp. 70–74, 2007.

[128] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[129] M. Monirul Kabir, M. Monirul Islam, and K. Murase, "A new wrapper feature selection approach using neural network," *Neurocomputing*, vol. 73, pp. 3273–3283, 2010.

[130] A. J. Mooij, "Constructing and reasoning about security protocols using invariants," *Electronic Notes in Theoretical Computer Science*, vol. 201, pp. 99–126, 2008.

[131] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," *Network, IEEE*, vol. 8, no. 3, pp. 26–41, 1994.

[132] S. Mukkamala and A. H. Sung, "Feature ranking and selection for intrusion detection systems using support vector machines," in *Proceedings of the Second Digital Forensic Research Workshop*. Citeseer, 2002, pp. 503–509.

[133] A. P. Muniyandi, R. Rajeswari, and R. Rajaram, "Network anomaly detection by cascading k-means clustering and c4.5 decision tree algorithm," *Procedia Engineering*, vol. 30, pp. 174–182, 2012.

[134] Y. L. Murphey and H. Guo, "Automatic feature selection-a hybrid statistical approach," in *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, vol. 2. IEEE, 2000, pp. 382–385.

[135] R. S. Naoum, N. A. Abid, and Z. N. Al-Sultani, "An enhanced resilient backpropagation artificial neural network for intrusion detection system," *International Journal of Computer Science & Network Security*, vol. 123, 2012.

[136] P. Ning, X. S. Wang, and S. Jajodia, "Modeling requests among cooperating intrusion detection systems," *Computer Communications*, vol. 23, pp. 1702–1715, 2000.

[137] M. R. Norouzian and S. Merati, "Classifying attacks in a network intrusion detection system based on artificial neural networks," in *Advanced Communication Technology (ICACT), 2011 13th International Conference on*, 2011, pp. 868–873.

[138] S. J. Oh and J. Y. Kim, "A hierarchical clustering algorithm for categorical sequence data," *Information Processing Letters*, vol. 91, no. 3, pp. 135–140, 2004.

[139] A. A. Olusola, A. S. Oladele, and D. O. Abosede, "Analysis of kdd'99 intrusion detection dataset for selection of relevance features," *Lecture Notes in Engineering & Computer Science*, vol. 2186, no. 1, pp. 162–168, 2010.

[140] H. Om and A. Kundu, "A hybrid system for reducing the false alarm rate of anomaly intrusion detection system," in *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*. IEEE, 2012, pp. 131–136.

[141] I. Onut and A. A. Ghorbani, "A feature classification scheme for network intrusion detection," *International Journal of Network Security*, vol. 5, no. 1, pp. 1–15, 2007.

[142] L. Ozdamar and O. Demir, "A hierarchical clustering and routing procedure for large scale disaster relief logistics planning," *Transportation Research Part E Logistics & Transportation Review*, vol. 48, no. 3, pp. 591–602, 2012.

[143] X. Pan, H. Gu, and Z. Zhao, "Feature selection framework research in extracting term definition," *Journal of Nanjing University of Aeronautics & Astronautics*, 2012.

[144] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: a review," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 90–105, 2004.

[145] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 25–41, 2013.

[146] N. Paulauskas and E. Garsva, "Computer system attack classification," *Electronics & Electrical Engineering*, 2006.

[147] ——, "Computer system attack classification," *Elektronika ir Elektrotechnika*, vol. 66, no. 2, pp. 84–87, 2015.

[148] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1226–1238, 2005.

[149] P. A. Porras and R. A. Kemmerer, "Penetration state transition analysis: A rule-based intrusion detection approach," in *Computer Security Applications Conference, 1992. Proceedings., Eighth Annual*, 1992, pp. 220–229.

[150] L. Portnoy, E. Eskin, and S. Stolfo, "Intrusion detection with unlabeled data using clustering," in *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001)*. Citeseer, 2001.

[151] Z. Ran, Q. Yao, X. Wang, and Y. Zou, "Application of pattern matching algorithm in intrusion detection technique," *Modern Electronics Technique*, 2009.

[152] R. M. Rimiru, G. Tan, and S. N. Njuki, "Towards automated intrusion response: A pamp-based approach," *International Journal of Artificial Intelligence & Expert Systems*, vol. 2, no. 2, 2011.

[153] M. Sabhnani and G. Serpen, "Application of machine learning algorithms to kdd intrusion detection dataset within misuse detection context." in *MLMTA*, 2003, pp. 209–215.

[154] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *bioinformatics*, vol. 23, no. 19, pp. 2507–2517, 2007.

[155] K. Sandeep and H. S. Eugene, "An application of pattern matching in intrusion detection," *Computer Science Technical Reports*, 1994.

[156] D. Schneider, "The state of network security," *Network Security*, vol. 2012, no. 2, pp. 14–20, 2012.

[157] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables," in *Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on*. IEEE, 2001, pp. 38–49.

[158] K. Shafi, "Online and adaptive signature learning for intrusion detection," *http://www.vdm-verlag.de*, 2009.

[159] S. Shan and V. Karthik., "An approach for automatic selection of relevance features in intrusion detection systems," 2011, pp. 215–219.

[160] H. E. Shao-Rong, "Research on application and design of ga in feature selection," *Computer Engineering & Applications*, vol. 46, no. 27, pp. 131–134, 2010.

[161] W. Siedlecki and J. Sklansky, "On automatic feature selection," *International Journal of Pattern Recognition & Artificial Intelligence*, vol. 2, no. 2, 2011.

[162] T. S. Sobh, "Anomaly detection based on hybrid artificial immune principles," *Information Management & Computer Security*, vol. volume 21, no. 4, pp. 288–314(27), 2013.

[163] J. M. Sotoca and F. Pla, "Supervised feature selection by clustering using conditional mutual information-based distances," *Pattern Recognition*, vol. 43, no. 6, pp. 2068–2081, 2010.

[164] T. M. Sven Ehlert, Dimitris Geneiatakis, "Survey of network security systems to counter sip-based denial-of-service attacks," *Computers & Security*, vol. 29, pp. 225–243, 2010.

[165] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, "Intrusion detection using fuzzy association rules," *Applied Soft Computing*, vol. 9, no. 2, pp. 462–469, 2009.

[166] K. M. Tan, K. S. Killourhy, and R. A. Maxion, "Undermining an anomaly-based intrusion detection system using common exploits," in *Recent Advances in Intrusion Detection*. Springer, 2002, pp. 54–73.

[167] M. Tavallaee, E. Bagheri, W. Lu, and A.-A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications*, 2009.

[168] D. Tian, J. Keane, and X. J. Zeng, "Evaluating the effect of rough set feature selection on the performance of decision trees," in *Granular Computing, 2006 IEEE International Conference on*, 2006, pp. 57–62.

[169] D. Turner, S. Entwisle, M. Denesiuk, M. Fossi, J. Blackbird, D. Mckinney, R. Bowes, N. Sullivan, P. Coogan, and C. Wueest, "Symantec internet security threat report," *Volume*, no. 5, pp. 277–278, 2007.

[170] J. Van Lunteren, "High-performance pattern-matching for intrusion detection," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, 2006, pp. 1–13.

[171] P. Veríssimo and L. Rodrigues, "Fundamental security concepts," in *Distributed Systems for System Architects*. Springer, 2001, pp. 377–393.

[172] G. Vigna, W. Robertson, and D. Balzarotti, "Testing network-based intrusion detection signatures using mutant exploits," in *Proceedings of the 11th ACM conference on Computer and communications security*. ACM, 2004, pp. 21–30.

[173] S. Vikas, R. Subrata, D. Dipti, E. Deniz, J. C. Principe, and N. Partha, "Feature selection in mlps and svms based on maximum output information," *IEEE Transactions on Neural Networks*, vol. 15, no. 4, pp. 937–48, 2004.

[174] L. T. Vinh, S. Lee, Y. T. Park, and B. J. D'Auriol, "A novel feature selection method based on normalized mutual information," *Applied Intelligence*, vol. 37, no. 1, pp. 100–120, 2012.

[175] D. Wagner and P. Soto, "Mimicry attacks on host-based intrusion detection systems," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, 2002, pp. 255–264.

[176] K. Wang and S. J. Stolfo, "Anomalous payload-based network intrusion detection," in *Recent Advances in Intrusion Detection*. Springer, 2004, pp. 203–222.

[177] N. Wattanapongsakorn, S. Srakaew, E. Wonghirunsombat, C. Sribavonmongkol, T. Junhom, P. Jongsubsook, and C. Charnsripinyo, "A practical network-based intrusion detection and prevention system," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. IEEE, 2012, pp. 209–214.

[178] X. Wen, "Development of a snort-based security network management and real-time intrusion detection system," *Journal of Beijing Normal University*, vol. 40, no. 1, pp. 40–43, 2004.

[179] L. Wenke and S. Salvatore, "Data mining approaches for intrusion detection," *Proceedings of Usenix Security Symposium*, vol. 16, no. 4, pp. 18–20, 1998.

[180] G. White and V. Pooch, "Cooperating security managers: Distributed intrusion detection systems," *Computers & Security*, vol. 15, no. 96, pp. 441–450, 1996.

[181] D. Wilson and D. Kaur, "Knowledge extraction from kdd'99 intrusion data using grammatical evolution," *Wseas Transactions on Information Science & Applications*, vol. 4, 2007.

[182] X. Wu, K. Yu, H. Wang, and W. Ding, "Online streaming feature selection," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 1159–1166.

[183] Y. L. Wu, C. Y. Tang, M. K. Hor, and P. F. Wu, "Feature selection using genetic algorithm and cluster validation," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2727–2732, 2011.

[184] C. Xenakis, C. Panos, and I. Stavrakakis, "A comparative evaluation of intrusion detection architectures for mobile ad hoc networks," *Computers & Security*, vol. 30, no. 1, pp. 63–80, 2011.

[185] T. Xia, G. Qu, S. Hariri, and M. S. Yousif, "An efficient network intrusion detection method based on information theory and genetic algorithm," in *Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International*, 2005, pp. 11–17.

[186] J. Xiao, "Design and implementation of maids-a distrbuted intrusion detection system based on mobile agent," *Computer Engineering & Applications*, 2003.

[187] J. Xu, A. Y. S. Lam, and V. O. K. Li, "A memory-efficient bit-split parallel string matching using pattern dividing for intrusion detection systems," *Parallel & Distributed Systems IEEE Transactions on*, vol. 22, no. 11, pp. 1904–1911, 2011.

[188] X. Xu, "Sequential anomaly detection based on temporal-difference learning: Principles, models and case studies," *Applied Soft Computing*, vol. 10, no. 3, pp. 859–867, 2010.

[189] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *ICML*, vol. 3, 2003, pp. 856–863.

[190] ——, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *International Conferences on Machine Learning*, 2003, pp. 856–863.

[191] D. Zhang, S. Chen, and Z. H. Zhou, "Constraint score: A new filter method for feature selection with pairwise constraints," *Pattern Recognition*, vol. 41, no. 5, pp. 1440–1451, 2008.

[192] Y. Zhang, L. Wang, W. Sun, R. C. Green, M. Alam *et al.*, "Distributed intrusion detection system in a multi-layer network architecture of smart grids," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 796–808, 2011.

[193] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 1151–1157.

[194] C. V. Zhou, C. Leckie, and S. Karunasekera, "A survey of coordinated attacks and collaborative intrusion detection," *Computers & Security*, vol. 29, pp. 124–140, 2010.

[195] Z. Zhu, Y.-S. Ong, and M. Dash, "Wrapper–filter feature selection algorithm using a memetic framework," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 1, pp. 70–76, 2007.