3-22-2016

Innovative Two-Stage Fuzzy Classification for Unknown Intrusion Detection

Xueyan Jing
sfan001@fiu.edu

DOI: 10.25148/etd.FIDC000288

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

INNOVATIVE TWO-STAGE FUZZY CLASSIFICATION FOR UNKNOWN INTRUSION DETECTION

A dissertation submitted in partial fulfillment of

the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL AND COMPUTER ENGINEERING

by

Xueyan Sharon Jing

2016

To: Interim Dean Ranu Jung
College of Engineering and Computing

This dissertation, written by Xueyan Sharon Jing, and entitled, Innovative Two-Stage Fuzzy Classification for Unknown Intrusion Detection, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

_____
Frank Urban

_____
Jean Andrian

_____
Deng Pan

_____
Hai Deng, Major Professor

Date of Defense: March 22, 2016.

The dissertation of Xueyan Sharon Jing is approved.

_____
Interim Dean Ranu Jung
College of Engineering and Computing

_____
Andrés G. Gil
Vice President for Research and Economic Development
And Dean of University Graduate School

Florida International University, 2016

ABSTRACT OF THE DISSERTATION

INNOVATIVE TWO-STAGE FUZZY CLASSIFICATION FOR UNKNOWN INTRUSION DETECTION

by

Xueyan Sharon Jing

Florida International University, 2016

Miami, Florida

Professor Hai Deng, Major Professor

Intrusion detection is the essential part of network security in combating against illegal network access or malicious cyberattacks. Due to the constantly evolving nature of cyber attacks, it has been a technical challenge for an intrusion detection system (IDS) to effectively recognize unknown attacks or known attacks with inadequate training data. Therefore in this dissertation work, an innovative two-stage classifier is developed for accurately and efficiently detecting both unknown attacks and known attacks with insufficient or inaccurate training information.

The novel two-stage fuzzy classification scheme is based on advanced machine learning techniques specifically for handling the ambiguity of traffic connections and network data. In the first stage of the classification, a fuzzy C-means (FCM) algorithm is employed to softly compute and optimize clustering centers of the training datasets with some degree of fuzziness counting for feature inaccuracy and ambiguity in the training data. Subsequently, a distance-weighted k-NN (k-nearest neighbors) classifier, combined with the Dempster-Shafer Theory (DST), is introduced to assess the belief functions and pignistic probabilities of the incoming data associated with each of known classes to

further address the data uncertainty issue in the cyberattack data. In the second stage of the proposed classification algorithm, a subsequent classification scheme is implemented based on the obtained pignistic probabilities and their entropy functions to determine if the input data are normal, one of the known attacks or an unknown attack. Secondly, to strengthen the robustness to attacks, we form the three-layer hierarchy ensemble classifier based on the FCM weighted k-NN DST classifier to have more precise inferences than those made by a single classifier. The proposed intrusion detection algorithm is evaluated through the application of the KDD'99 datasets and their variants containing known and unknown attacks. The experimental results show that the new two-stage fuzzy KNN-DST classifier outperforms other well-known classifiers in intrusion detection and is especially effective in detecting unknown attacks.

TABLE OF CONTENTS

TABLE OF TABLES

TABLE PAGE

TABLE OF FIGURES

FIGURE PAGE

CHAPTER 1

INTRODUCTION

The security-related issues of computers and networks have become exceptionally significant. The rapidly growing use of computer systems and the internet has emphasized the need to guaranteeing secure accessibility of billions of computer users to internet information. How to insure legitimate access to privileged information has been a major area of concern for information processing technology in computer networks. Defense mechanisms against malicious attacks must be able to protect the network before the system becomes compromised [1].

The protection policies being investigated since the last three decades, to date, have also addressed the problem of information security. Security denotes the property of protection against compromised: unauthorized dissemination of information. The security policy defines access domains of subjects based on considerations derived from the manufacturers of computer systems. Detection mechanisms are passive policy enforcement devices that are different with prevention mechanisms. Prevention mechanisms attempt to intercept potential violations, detection mechanisms monitor system activities, often maintaining records to aid in damage assessment, limitation, and recovery. However, a detection mechanism may include only very simple journaling programs that have no logic whatsoever regarding the significance of the events they record. Conceivably, certain violations could only be recognized after the fact through complex logical and statistical analyses.

One method of a computer security mechanism is intrusion detection. Its purpose is to monitor and identify malicious behavior, unauthorized access, or any other types of unfaithful forces that attempt to access information. On the other hand, if any attacks occur, intrusion detection should have the potential to undermine any safeguards that might have been built into computer programs or application systems. An intrusion detection system [2] (IDS) was proposed to provide maximum protection to systems with sensitive information. Unlike a traditional security mechanism such as firewalls, time-of-use stamps, audit trails, flow control, or authentication, IDS monitors these detection systems for both malicious and normal connections, which make IDS an imperative player in the computer network security process.

Cyber-crimes have happened seriously and continuously in many computer networking, because the attackers keep on developing new forms of attacks where current IDS tools failed to detect or stop these new attacks, and also the patterns of normal traffic will be changed.

1.1 General Approaches to intrusion detection

About fifty years ago, IDS was first brought out as a security mechanism for monitoring and deciding the allowability of all access by information processors to information repositories. An IDS must satisfy the following properties:

- It is wide-ranging: all computer network access from internal or external, both novel or known malicious intrusion are monitored and enforced
- It is safeguarded: the function of IDS may not be maliciously or accidently modified by unauthorized forces

- Its well-timed: IDS may analysis and make a decision in a timely manner to respond for the abnormal behaviors.
- It has provably proper performance: IDS must faithfully enforce the specified protection strategy with higher detection rate and lower error rate.

Basically, there are mainly two types of IDS techniques: anomaly detection [5] and misuse detection [6] techniques.

Misuse method builds up the attack signature and looks for known attacks. Misuse detection model compares the incoming traffic connections with previously stored known attack patterns. If there is deviation that exists then these connections are declared as attacks. The primary modes of building up abnormal profiles are auditing and surveillance. Auditing keeps the records of activities include users logging in and out, the granting and revoking of access rights, and access violations, both attempted and successful. Surveillance is the active monitoring of both normal and abnormal activities on the system in real-time and useful to commands in charge of security. The majority techniques in IDS are using this type of detection. The disadvantage of misuse detection is that it can only detect known attacks and fail to detect unknown attacks. Another shortcoming of misuse detection is that it spends time on creating the profile of the new attack signature and updates it manually into the IDS model.

While anomaly methods build up normal patterns and can detect abnormal access that deviates from the normal patterns in profiles. Whenever a possible mismatch happens, it implies that unauthorized intruders attack and sneak into the computer system. These normal pattern profiles are built through any computer system behaviors, such as access command lines, data audit logs, network packets, and thread calls, etc. Data mining

techniques and machine learning methods are often used to help to identify the network connection as normal connection or malicious attacks. Anomaly techniques can indicate a potential threat both known and unknown whenever there is a significant deviation from anomaly model of reference. Anomaly techniques also have their disadvantages. It suffers from the high false alarms generated from the IDS models by misclassification normal behavior into attacks [7]. Both of these methods aims to provide the solution that can offer more accurate detection rate while keeping low false positive rate.

Numerous researches have been realized in IDS. Some classical IDS methods like decision trees and variants of Bayes are adopted to detect attacks in computer network connections, and still have the problem of poor detection performance to classify malicious attacks (especially unknown attacks) from normal traffic connections.

1.2 Problem Statement

No matter using which IDS methods, many challenging issues have grown in designing IDS models. Both misuse detection and anomaly detection are suffering from these issues:

1.2.1 Feature Selection

A large amount of computer network data needs to be collected for analysis to identify the abnormal behaviors from normal behaviors. These collected network data is described with many features such as protocol type, connection duration, destination port#, source port#, and other features. Thus in turn, the large dimension of these features makes the IDS process more difficult and complex with higher error rates. Some features are irrelevant and redundant to the attack patterns and it could slow the

identification process. Therefore, Using feature selection or feature extraction is necessary to the performance of the detection techniques adopted.

1.2.2 Supervised Learning and Unsupervised Learning

In supervised learning the algorithm is trained with predefined opinions based on labeled data to help with predictive model testing. In some way, supervised learning is similar with learning with experts because the answer (labels in IDS) is offered. The unsupervised learning scenario would not use the labels at all to help IDS model understand the data, also similar with learning without experts because the answer is not offered. Supervised learning costs time, and require experts and measurements, while unsupervised learning function well with noisy data containing corrupt data that cannot be analysis or interpreted correctly. Semi-supervised learning is a method that employs both labeled (usually small amount) and unlabeled data (usually large amount) for training. Semi-supervised learning sometimes helps predictive model testing to reduce the cost and improve the performance.

1.2.3 Ambiguity and Uncertainty Classification

Many researches in IDS have been dedicated to solve the ambiguity of traffic data and need to be able to acknowledge and handle with ambiguity. That is, to some degree the connections are too ambiguous to be assigned to malicious classes or normal class. It is difficult to completely distinguish certain types of attacks, which often embedded in the packets and is hard to be separated from normal connections. The problem could be the patterns of these types of attacks is alike the patterns of normal connections or the boundary between normal connection and attack are blur. We really hope to find out how

ambiguity issues related to network data effect the security of computer systems and to offer a better interpretation of ambiguity as a concept of fuzziness in IDS, which can help researchers to deal with the ambiguity problem.

1.2.4 Unknown Attack Detection

Cybercrimes have happened seriously and continuously in many computer-networking systems, because the attackers keep on developing new form of attacks. The malicious unknown attacks arise from the requirements for open use and sharing information system. Present day computer systems require largely open sharing systems. The major threat to these systems is that of external penetration. In the case of open use and sharing systems there is an implication of unprotected communications lines for subscribers not performing classified processing that increases the exposure to outside penetration. The external penetration threat could be countered by using combinations of different communication security techniques, but still missed in many cases. These techniques, some highly advanced, are the bulk of the present state-of-the-art in computer security.

The technical issue of novel attack detection is concerned with the concept of unknown attack classification methods. By this we recognized that the nature of shared use computer systems present to a malicious users a unique opportunity for attempting to subvert through programming the mechanism upon which security depends. In effect, the defense against new types of attacks should surround the system and its user environment with a solid barrier that must be breached before the system can be compromised.

While we emphasize the threat both known and unknown from a malicious users, we are not unmindful of other security threats and risks. The problems of accidental spillage of

classified information, physical penetration of system sites, interference with or intercept of communications, mishandling of classified material and the like are serious. To a large extent, these problems are common to any information system processing classified information including mobile network and wireless sensor works, and can be solved by well-understood techniques. However, the hackers in the context of a resource shared system always presents new type of threats, control of which is necessary before the objective of full use of shred computer systems can be realized.

1.2.5 Penetration of systems

There is little question that contemporary commercially available systems do not provide an adequate defense against malicious threat. Although current IDS models have improve the detection rate and lower the error rate, most of these systems are known to have serious design and implementation flaws that can be exploited by individuals with programming access to the system. As an instance of this, we note that the current computer system has a number of major flaws that would permit hackers to subvert the nominal security controls that exist in the system. The IDS designs still face the challenges of implementation flaws in most contemporary systems permit intruders to seize unauthorized control of the system, and thus have access to any of the information on the system. While the techniques for defense the vulnerability on contemporary systems vary, they ultimately boil down to graining either directly or indirectly, an authorized access capability to classified data.

1.2.6 Operational Cost and Economic Consideration

Each IDS method the consequences of the inadequate security mechanisms in current computer network systems are both the potential for loss of information critical to information security by malicious attacks and higher cost of operation. It means how effective or in what degree the IDS models can protect the system and identify malicious attacks from traffic connections. Operational requirements for intrusion detection mechanism are based on the need for rapid access to and dynamic sharing of information. At present, these requirements cannot be met without great risk of penetration and compromise. Higher costs of operation include costs due to separate computers for separate applications, restricting use of remote terminals, costs of physical protection of remote terminals and associated crypto devices, and the costs of clearing all user personnel to the highest level of classified information processed by a system. Pursuing the methods recommended in IDS will have significant effect in reducing these costs perhaps yielding a high reduction of the cost of network systems that handle the classified data.

It would be simpler if the single user had complete control over his processing environment, including his data and programs. After a few years users began demanding better utilization of the resources. The response to this demand for more efficiency gave birth to multiplexing techniques,

1.3 Research Hypotheses

- Ambiguity and uncertainty problem can be solved by Fuzzy C-means (FCM) weighted k-NN reasoning *Dempster-Shafer* theory (DST) method.

- Unknown malicious attacks can be correctly classify by entropy function to improve the overall detection rate involving unknown abnormal behaviors in discovering intrusive behaviors.
- Semi-supervised learning is used to help predictive model testing to reduce the cost and improve the performance by using both labeled data (usually small amount) and unlabeled data (usually large amount) for training.

1.4 Proposed Approach

We develop a novel soft computing method based on the fuzzy belief [21] in this paper for the classification of computer network traffic connections to detect unknown attacks efficiently and effectively in the IDS process. Our work is based on the fuzzy belief IDS structure, which combines the fuzzy set theory and DS theory to search for intrusive behaviors in network traffic connections, and then classify the intrusive behaviors into normal class or attack class. Nevertheless, the fuzzy belief classifier using combination of evidence under fuzzy environments can only recognize the normal and attack traffic connections that previously collected in the training data set.

To overcome this drawback, we developed Fuzzy C-means (FCM) weighted k-NN reasoning DST classifier for detection of previously unknown intrusions when the available information is imperfect and ambiguous to be specifically defined as normal or attacks. One novel aspect of this classifier is that it can correctly classify novel malicious attacks by entropy function to improve the overall detection rate involving unknown abnormal behaviors in discovering intrusive behaviors. In addition, the other novel aspect of this classifier is that the decision rules generated by fuzzy reasoning weighted k-NN

method for probability assignment of focal element are treated as evidences which strengthen the deviation knowledge of unknown attacks from normal instances and known attacks. The combination of all the fuzzy k-NN focal elements is then combined using the generalized Dempster's rule. The FCM rules generation process is similar to [14] and this classifier is to classify a network traffic connection into three categorize of normal access, one of known attacks or unknown attacks.

1.5 Dissertation Organization

The dissertation is organized as follows.

The dissertation is organized as follows. Chapter II we review existing literature related to the issues involved in implementing IDS in computer networks. We begin by discussing evolutions of Intrusion Detection Mechanisms then followed by the current techniques used to improve the performance of IDS design of computer networks.

Chapter III initially explains we propose fuzzy belief k-nearest neighbors (k-NN) classifier to solve intrusion detection uncertainty problems caused by ambiguous and limited data. This single classifier incorporates fuzzy clustering technique along with Dempster-Shafer theory into our intrusion detection scheme to handle the ambiguity of traffic connections. Also, the k-NN technique is applied to further speed up the intrusion detection process.

Chapter IV introduces the DARPA KDD 99 data sets as our benchmark in our experiment to present the comparison results.

Chapter V we propose an innovative fuzzy classifier for effectively detecting both unknown attacks and known attacks with insufficient or inaccurate training information.

Chapter VI we improve the overall network intrusion detection rate by proposing and using an innovative three-layer hierarchy multi-classifier detection scheme called ensemble classifier.

Chapter VII draws the conclusions, presents the contribution of this dissertation, and lists future research directions.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we review existing literature related to the issues involved in implementing intrusion detection systems (IDS) in computer networks. We begin by discussing evolutions of Intrusion Detection Mechanisms in Section 2.1. The current techniques used to improve the performance of IDS design of computer networks are described in details in Section 2.2.

2.1 Evolutions of Intrusion Detection Mechanisms

Because the problem of information security in computer-based systems became visible only with the development of and acceptance of resource sharing systems, there is long history of previous work started in 1967 on the Defense Science Board Task Force. The computer security was first convened that this Task Force would analyze the problem and recommends a research and development program that would provide solutions to the extant problems of that time. During the course of that work it was discovered that the problem was not well understood and as a consequence the final report prepared by the Task Force contained less in the way of a recommended R&D program than had originally been thought possible. The report did, however contain and extensive discussion of the scope of the problem as well as definitions of terminology that were sadly lacking at that time.

There has been some efforts made by “tiger team” that have expended a moderate amount of energy in demonstrating the security inadequacy of both standard commercial systems and those modified to provide security controls. The value of “tiger teams” in testing

computer security is questionable because the results of the effort are highly dependent on the quality and experience of the personnel assigned to the teams. Even if corrections are made as a result of flaws found by a team, there is no assurance that all flaws have been found and corrected. The activities of the tiger team can only reveal system flaws and provide no basis for asserting that a system is secure in the event their efforts are unsuccessful. In the latter event, the only thing that can be stated is that the security state of the system is unknown.

There has been some efforts made by “tiger team” that have expended a moderate amount of energy in demonstrating the security inadequacy of both standard commercial systems and those modified to provide security controls. The value of “tiger teams” in testing computer security is questionable because the results of the effort are highly dependent on the quality and experience of the personnel assigned to the teams. Even if corrections are made as a result of flaws found by a team, there is no assurance that all flaws have been found and corrected. The activities of the tiger team can only reveal system flaws and provide no basis for asserting that a system is secure in the event their efforts are unsuccessful. In the latter event, the only thing that can be stated is that the security state of the system is unknown.

Various members of the Defense and Intelligence communities have funded a number of independent projects concerned with various aspects of computer security. In addition, a fairly major effort to provide security controls to a system that existed within a benign environment in the Intelligence community has taken place over the past four decades. While these controls are of interest and provide a certain degree of implementation of

security procedures, they did not address the question of providing technical security against malicious attack.

Given the problems of current hardware and operating systems some users (SFGWC, AFLC) have been driven to the development of large software packages that mediate between applications programs and operating systems. Such packages are capable of providing a degree of security in a benign environment but exact a very large price for storage space and execution time. These packages seem to offer little protection against a hostile programmer or possible underlying trapdoors and may be employed to protect a small amount of classified data. Thus, their cost-effectiveness, at least, is subject to question.

Anderson stated in 1980 that security audit trials can plan an important role in a security program for a computer system. As audit trails are presently structured on most machines, they are only useful primarily in detecting unauthorized access to files. For those computers, which have no access control mechanisms built into the primary operating systems, the audit trail bears the burden of detecting unauthorized access to system resources. As access control mechanisms are installed in the operating system, the need for security audit trail data will be even greater. It will not only be able to record attempted unauthorized access, but will be virtually the only method by which user actions which are authorized but excessive can be detected.

In 1982, the Advance Research Projects Agency (ARPA) has funded work at Rand Corporation, Information Systems Institute (USC) and Livermore Research Laboratories to analyze the security adequacy of selected commercial operating systems and to develop methodologies of security assurance. These programs have not been sufficiently

developed to provide any assessment of this potential contribution to the solution of some of the problems perceived by the computer security requirement at that time. Finally, the problem of computer security achieved major recognition from IBM's announcement of their intention to spend 40 million dollars on the problem and it directed to the enhancement of IBM product, Resource Security System (RSS).

Denning in 1987 proposed an intrusion detection model based on the hypothesis that the vulnerabilities of computer include abnormal behavior of authorized user. This represents a milestone in the research in the area of intrusion detection model. The model is rule-based pattern matching model with the basic idea of monitoring the standard behavior on a computer system and searching for deviations in all the behaviors. There are no special features in this model. This IDES model can detect malicious attacks in real time with better performance.

A trend pointed out with considerable emphasis by the Requirements Working Group is the movement towards the establishment of large dispersed networks of related computer systems. A F Global Weather Center, for example, will interconnect several of its own computer systems. In addition, this interconnected complex will be tied to other weather processing centers and to the command control system of SAC and MAC. SAC plans to tie several command control computers together, and may also interface intelligence processing systems. The MAC command control system, MACIMS, will be implemented as a network of WWMCCS computes. Plans are being formulated for a network to interconnect all of the WWMCCS computer installations. As networks of the types mentioned are developed, computer security problems, already difficult, become much more complex. For example, there is a possibility of untrustworthy processor in a

network collecting classified data from other processors by making apparently legitimate requests. Computer networks that have one or more nodes that can be accessed by users with clearances below the highest level of information in the network constitute multilevel networks. The security threat posed by such operation is that in general the compute to computer communications are accepted as valid on the questionable basis that the other computer has a high security reliability. However, if a malicious user can exercise control of a node, the entire network may be compromised. In a network, it is essential that there be reliable security controls, that the nature of these be understood, and the network does not inadvertently provide the means to bypass these controls. While there are growing requirements for interconnecting computer systems into networks the dimensions of the security problem are unknown. Much more information is needed on both the networks and their security requirements.

2.2 Intrusion Detection Techniques

In research area, there are many classifying algorithms used in IDS models. These techniques are typically based on the naïve Bayesian method [6,7], Support Vector Machine [8-11], Particle Swarm Optimization [12,13], Generic Algorithm [14], neural networks [15-18], k-nearest neighbor (KNN) methods [19,20], fuzzy c-means (FCM) methods [21,22], Dempster-Shafer Theory Of Evidence [23-26] or other decision-tree based ad-hoc methods [27]. These techniques are introduced in details in the following section. In particular, these techniques are developed as classifiers, which are used to classify or recognize whether the incoming Internet access is the normal access or an attack.

2.2.1 Naïve Bayesian

The Naïve Bayesian classifier is based on conditional probabilistic to perform decision of a classification problem. It uses Bayes' Theorem [1] with independence assumptions, which assumes a set of features are conditionally independent of one another given a class. In this case, the Naïve Bayes classifier has only one single parent node and several children nodes with unknown class labels, and these nodes represent each variables or attributes in data. When a set of classes are observed in the training data, the naive bayes classifier then assign an observed data to one of classes with highest probability. By applying Naïve Bayesian classifier to an intrusion detection task, a set of training network traffic data is given to find the prior probabilities for normal or a known class of attacks. As unseen network traffic arrives, the classifier then uses Bayes Theorem to decide which class the traffic should belong to.

There are basically two types of classification model, the most widely used methods are: Decision trees model and Naïve Bayesian Model, NBC). The Naïve Bayesian Model is stemmed from the classical mathematics with strong mathematics background and stable classification results. At the same time, NBC needs less parameter and is less sensitive to insufficient and incomplete data. Theoretically, NBC has lower error rate than other classifiers. Yet in reality, Domingos and Pazzani [2] found NBC suffers from the consumption of each attribute is independent from each other. This consumption is impossible in real life which effects the classification performance results in some degree. Under the circumstances of high dimensional attributes and more correlation attributes, NBC tends to have less detection rate than Decision trees classifier.

Kononenko [3] developed Semi-Naïve Bayesian classifier by dividing all the attributes into different groups where attributes in one group have no correlation to other groups. This means the attribute in one group is less independent to one on other groups. He applied this classifier in 4 medical domains. In two domains, the performance of Semi-Naïve Bayesian classifier has similar output with Naïve Bayesian classifier, and in other two domains, the Semi-Naïve Bayesian has better performance than Naïve Bayesian.

Langley and Sage [4] decribed a revised algorithm called the selective Bayesian Classifier especially to solve the problem of high correlation data. This algorithm only select certain features to use for the final decision making process. The forward selection method is used to modify the data by deleting less informative or redundant features to improve the detection rate, lower the error rate, and at the same time better worst-case time complexity.

2.2.2 Neural Networks

A backpropagation neural network uses a feedforward structure to solve classification problems by its supervised learning algorithm. It consists of a collection of processing units that are highly interconnected, and it also consists of one hidden layer like black box [5-9]. Tamura [10] stated that four-layered feedforward neural network with two hidden layers will reduce the hidden neurons to (N/2)+3 given N input-target relations exactly. He approved that four-layered feedforward neural network performs better than the three-layered feedforward network with less neurons and error rate.

Razavi [11] provides a framework with computational budget dependent to mimic the real world computational optimization problem for testing the algorithm of Neural

Networks. For the purpose of efficient and effective optimization, using more than one hidden layer in Neural Networks could be confusing, interference and time consuming. The single hidden layer Neural networks could clearly demonstrate that an efficient performance over multi layer Neural Networks without metamodels.

The network weights are updated by using gradient-based optimization algorithm during the training period [12-14]. Single layer Neural Networks can improve the classification performance with exceptionally training speed, because the training in Neural Networks requires complex optimization with local minima. When the network converges to the local minima of error, the output layer of the network will show the result when data is fed into the input layer.

Based on the data given for training, neural networks has the ability to learn how to process intrusion detection tasks. It acts as a computational model to process the network traffic information. By the use of training procedure, the neural network gains the knowledge to extract the normal and attack signatures from the provided data automatically. With its ability to generalize from learned data, the neural network performs generalization of attacks and fault tolerance to imprecise and uncertain information. At the end of the training procedure, the future network traffic are then identified as whether malicious attacks or normal usage behavior.

2.2.3 Fuzzy Logics

Fuzzy logics mimicking the human brain show a new way of inference and determination, and it tries to describe the uncertainty of some models to explain the powerful non-linear objects. in 1965, Zadeh [15]described fuzzy sets as new concepts for fuzzy logics in

information and control area. Fuzzy sets allow each element belong to more than one set of normal or attack with a belonging membership degree which breaks the traditional crisp binary belongings “yes” or “no”. It is also used in the alarm generation process to reduce the false alarms. The membership degree is continuous value between 0 and 1, when the membership degree is 0 or 1 the fuzzy set has only two elements as the classical binary sets.

In 1974 Mamdani [16] applied fuzzy logics on the control of steam machine and pot, which validate the performance of fuzzy control. In 1977 Pappis [17] applied fuzzy logics into traffic control on cross road transportation area, and it reduced the average waiting car period by seven percent.

Owen [18] created fuzzy rules to perform fuzzy logics as an adaptive expert intrusion detection system in computer networks. He categorized the input features as five membership grades: LOW, MEDIUM-LOW, MEDIUM, MEDIUM-HIGH, and HIGH, which are represented by numeric values. There are many ways to define fuzzy rules, for example

IF condition THEN consequent

A probabilistic model was founded by Hooper’s using rule of combination of evidence as a non-Bayesian approach. This neutral zone classifier is very similar to the intuitionistic fuzzy logics (IFL) in concept [19], and the three partition including truth, falsehood, and indeterminacy first mentioned by J. H. Lambert [20] with the credibility of one evidence influenced by the opposite evidence of another. Then it was further investigated by Lukasiewicz to divide it into 0. ½, and 1 values. Koopman [21] originally presented the concept of upper and lower probability, and was followed by Shafer (1976) [22], who

extended it to the Dempster-Shafer Theory of Belief Functions (DST) by proposing the Belief and Plausibility equations and applying the rule to combine two evidences. Lambert states three chances: chance p of accurate, chance q of mendacious and chance 1-p-q of careless, whereas three components: accurate, mendacious, careless sum up to 1. Fuzzy logics with infinite levels between the intervals of [0, 1] are introduced by Zadeh [23].

2.2.4 K-Nearest Neighbor

Non-parametric methods such as the voting k-nearest neighbor (k-NN) rule [3] have been used. However, one of the problems encountered when using this rule is the challenge of handling uncertainty due to insufficient and incomplete knowledge in identify network traffic into the normal or abnormal patterns. Due to the uncertainty of intrusive belief value, the use of common approaches can potentially limit the capability of these techniques. For this reason, Keller [8] have incorporated some concepts of fuzzy sets theory into the voting k-NN procedure, and developed a fuzzy k-NN rule which has been used heavily for many years. In his fuzzy k-NN algorithm, the membership in each class of a traffic connection to be classified is determined by combining the membership values of its k nearest neighbors rather than a crisp class as in voting k-NN. In addition of ensuring more informative content of the classification results, this rule yields a lower error rate than voting k-NN.

The k-NN classifier is simple but effective in many pattern classification applications. For an input to be classified, a number of k nearest training patterns are obtained based on the Euclidean distance measurement between the input and every training pattern. The

input is then simply assigned to the class by majority voting, i.e., the input is classified to the most frequent class label among the k nearest training patterns. However, a major drawback of k-NN algorithm is that the precision of classification may decrease if all selected k nearest training patterns are equally important without considering the differences of distances [7]. Furthermore, while processing an intrusion detection task, some of the intrusive patterns are similar to those of normal activities. The boundaries between those attacks and the normal behavior are always unclear. To eliminate this drawback, fuzzy k-NN classifier is proposed and fuzziness is introduced into it. It assigns multiple membership grades to classes rather than a single class by the use of the distance differences from the k nearest training patterns. The confidence values are in proportion to the correspondent membership grades that the input network traffic belongs to certain classes

2.2.5 Dempster-Shafer Theory

The Dempster-Shafter Theory (DST) [9] [10] [11] [12] has been used to resolve uncertainty of information in a decision process. In the combination process, data fusion and evidential reasoning methods have been used to achieve a better performance [3]. In [9], the classification problem was addressed from the point of view of the Dempster-Shafer theory of evidence. In this approach, each of the k nearest neighbors of a pattern to be classified is considered as an item of evidence supporting certain hypotheses concerning the class membership of that pattern. This information is provided in the form of a belief structure, defined as a function of the distance between the pattern under consideration and its neighbor. In [13], Shafer's theory was developed under the

condition that each belief structure mass function is defined over fuzzy subsets instead of crisp subsets. Yager [13] represented the evidence of each training pattern by a belief structure with fuzzy focal elements. The combination of the different structures is then performed using the generalized Dempster's rule. Thus, if the the available information is imperfect and ambiguous, soft computing techniques such as evidential reasoning, and fuzzy logic must be adapted to handle such uncertainty and imprecision in classification process. To a large extent, intrusions are common to any computer network systems processing classified information and may be solved by well-understood techniques [4]

2.2.6 Multiple Classifiers Systems

Besides the notability of multiplicity among the base classifiers, the right choice of a combination method is also an important issue in creating a supreme performance. A variety of combination methods have been reported for combining the outputs of the base classifiers into an ensemble result. According to their characteristics, they can be classified as linear combination methods, non-linear methods, statistical-based methods, and computationally intelligent methods. Linear combination method is the simplest method to fuse base classifiers' outputs together. Summation and average are the popular ways for the combination. Non-linear method such as majority voting is used when the output of classifier is ranked list of classes in accordance with the degree of belief on classes the input pattern belongs to.

There are three topologies for ensemble classifier: cascading ensemble classifier, parallel ensemble classifier, and hierarchical ensemble classifier as shown in figure. The cascading ensemble classifier in figure 2.1 will pass on the first classifier result to the

second classifier to form the final output. The data flow looks like a chain in this cascading process.

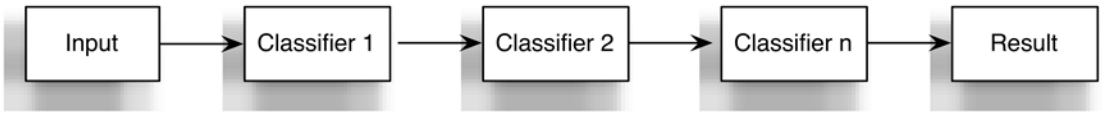

FIGURE 2.1 LAYOUT OF CASCADING STRUCTURE OF ENSEMBLE CLASSIFIER

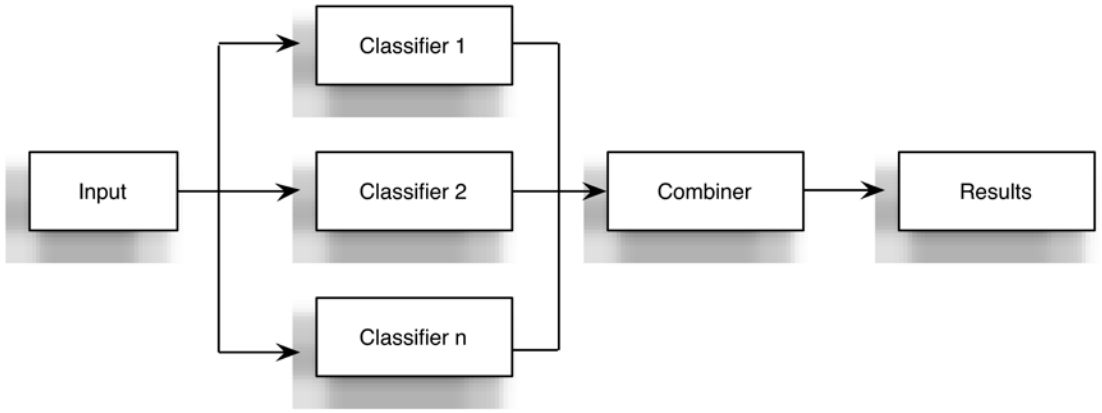

FIGURE 2.2 LAYOUT OF PARALLEL STRUCTURE OF ENSEMBLE CLASSIFIER

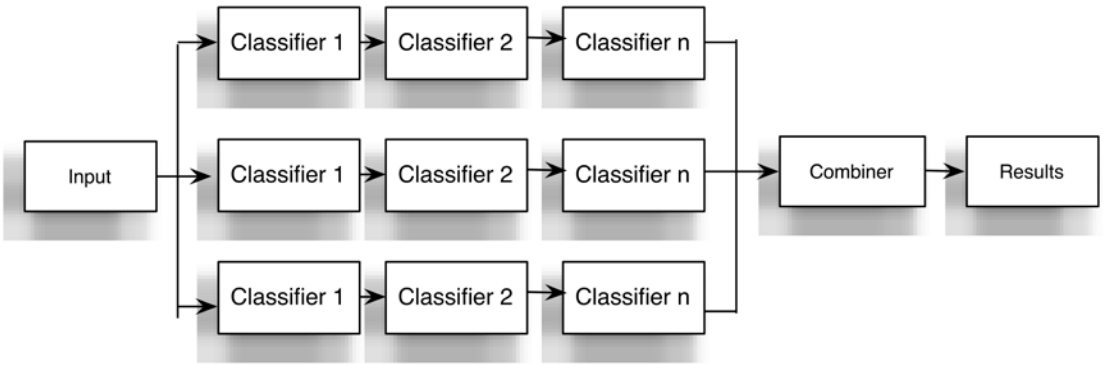

FIGURE 2.3 LAYOUT OF HIERACHICAL STRUCTURE OF ENSEMBLE CLASSIFIER

Usually it is very difficult to select the second classifier to compensate the errors produced by the first classifier. The parallel ensemble classifier in figure 2.2 is formed

by several paralleled base classifier and a combination method. The final result is produced by combining these base classifier outputs through the combination method. Selecting the base classifier is very important in this parallel structure model so that each base classifier should compensate each other's output. A careful delicate combination method will improve the detection performance greatly in ensemble classifier. The hierarchical ensemble classifier in figure 2.3 is basically the combination of cascading and parallel structure.

CHAPTER 3

SINGLE CLASSIFIER FOR INTRUSION DETECTION

In this chapter, we propose fuzzy belief k-nearest neighbors (k-NN) classifier to solve intrusion detection uncertainty problems caused by ambiguous and limited data. This single classifier incorporates fuzzy clustering technique along with Dempster-Shafer theory into our intrusion detection scheme to handle the ambiguity of traffic connections. Also, the k-NN technique is applied to further speed up the intrusion detection process.

First, we introduce uncertainty problems that exist in the IDS process and its influence on the IDS design. Then we describe the integration of the fuzzy-C-mean clustering techniques by grouping similar traffic connections together, and D-S theory by combining evidences from class labels and distances of its *k* nearest neighbor pairs into our proposed single classifier to solve the uncertainty problem.

3.1 Reasoning about Uncertainty is a Necessity

Uncertainty exists in every event and happens unpredictably. For example, the police department does not get involved with every residence's personal life. Whenever there are accidents or criminal events happened, the corresponding department or police officer would start to investigate the events or outlaws. That is to say, the accidents or criminal events only happen occasionally and unpredictably in some of the places in the world. Police officer could pay more attention to certain suspects and judge the criminal committers by his appearance or actions. With the similar function of the police department, the intrusion detection mechanism would have to deal with uncertainty caused accordingly by the computer activities. The IDS identify the traffic connection as

normal or abnormal activities by the symptoms and effects of these activities. It could happen that the IDS identify the attack as normal activities by mistakes when there is no clear distinction between normal and abnormal activities for a computer user. When the IDS identify the connection as normal or attack activities, the uncertainties exit in every step of the detection process.

There are two types of uncertainty: aleatory and epistemic. Aleatory means the uncertainty that inherently exists in events and is stochastically irreducible. While epistemic uncertainty happened due to the fact of lack of enough information of the system or environment. If more information or knowledge is collected about the system, it is possible to reduce the uncertainty degree. The problem of uncertainty in intrusion detection could be both of these types. There is no clear distinction between normal and abnormal activities for a computer user because sometimes activities seem similar to those of normal activities. Besides, there are inadequate and limited information observed by the monitoring tools and detection mechanisms.

3.2 Fuzzy Belief k-NN Classifier Modules

The machine learning techniques are just trying to copy the human brain by the machines. Classification becomes very important steps in machine learning techniques, whose goal is to identify which part of the data are normal or attacks by using advanced algorithm to classify the crucial information automatically into two groups in this chapter. The proposed fuzzy belief k-NN classifier is to identify which traffic connections are normal connections or intrusion attacks.

By choosing the right methods to get the data subsets can greatly decrease the classification time and improve the detection rate of the malicious attacks. During selecting the data subsets, the decision rules are generated from the incoming traffic data sets. These rules are also used later on to the classification phase to help to test the selected data from the attacks. Then we map the result features of the data fusion steps of final decision.

There are basically four models are described in the proposed ID platform. Through these models we can detect the incoming traffic from malicious attacks. The incoming datasets we use the DARPA KDD 99 to mimic the normal activities and abnormal activities. There are so many different kinds of data sets in the whole world. But how to select the right purpose oriented datasets is the main task in the intrusion aware architecture. The following step would be how to abstract the features from all these interrelated data sets. In other words, the subsets of data play an important role for the following classification phase.

Intrusion detection in fact is a classification task. In our work, the goal is to identify Denial of Service (DoS) attacks, Probe attacks, User to Root (U2R) attacks, and Remote to Local (R2L) attacks from the intrusion detection benchmark data set, i.e. *DARPA Intrusion Detection Evaluation data set KDD99*. This chapter applies an intrusion detection technique, called fuzzy belief k-nearest neighbors (k-NN) anomaly detection by using k-NN technique to reduce the computation time in application to the computer networking system. The approach uses a combination of fuzzy clustering technique and Dempster-Shafer theory. Since both of them have merit of resolving the uncertainty problems caused by limited and ambiguous information during a decision process.

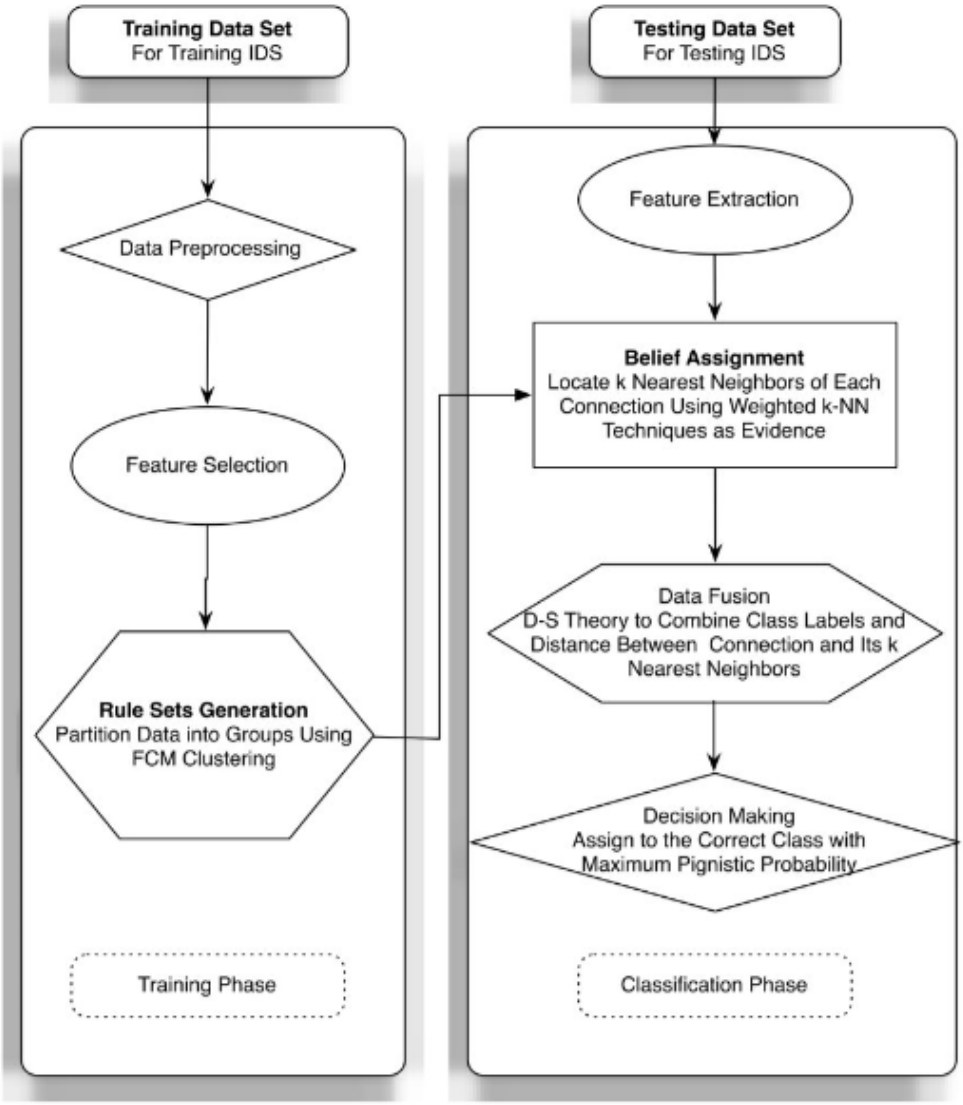

FIGURE 3.1 PROPOSED INTRUSION DETECTION STEPS

There are two phases in our intrusion detection system: training phase and classification phase shown in figure 3.1.

In the training phase, the fuzzy decision rules are generated from the training data. Then in the classification phase the network connection would be classified as normal or attacks based on this decision rule. Figure 3.2 shows the general four modules of the architecture of the proposed system.

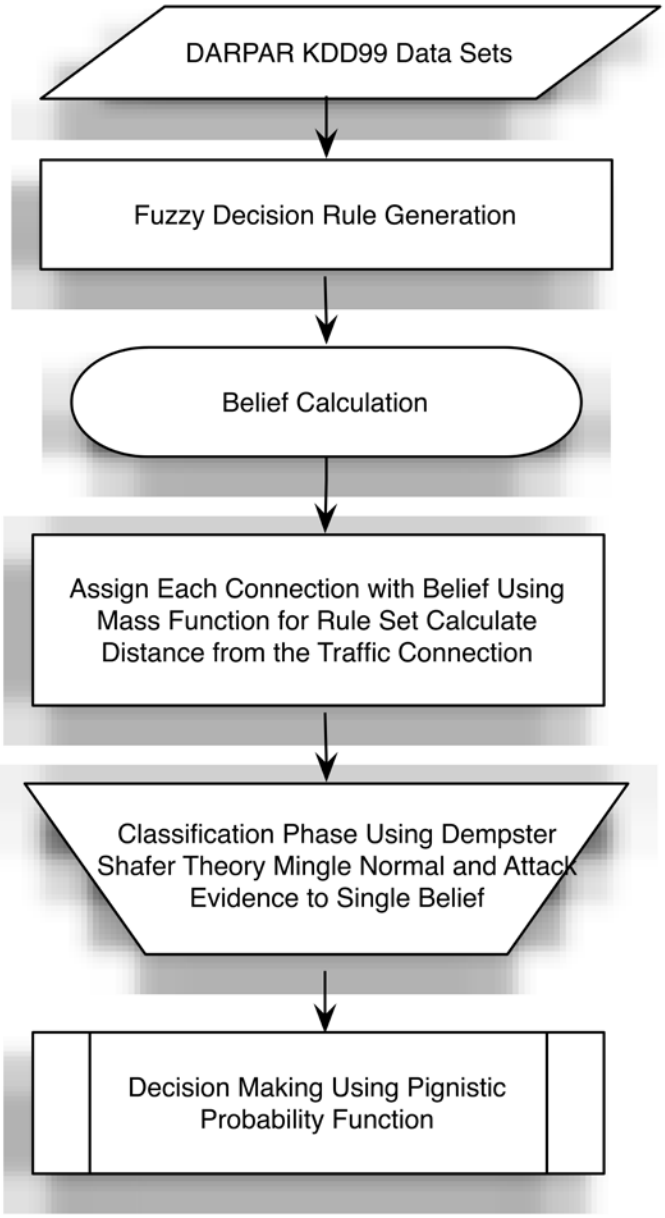

Figure 3.2 FOUR MODULS

- The first module is the Fuzzy Decision Rule module. The fuzzy decision rules are generated from the training data by using fuzzy c mean clustering algorithm. This can be used in the future work in the classification phase.

- The second module is Calculating Belief module. By calculating belief of each sub data sets, we assign each incoming traffic with the belief results using Mass function for each rule set. Also the distance from all the traffic connection in the training data sets are calculated by using k-NN rules.
- The third module is the Classification module where Dempster Shafer Theory is used. It computes the probability to automatically extract evidences including normal or attack class from training network traffic data. Then we mingle the normal evidence and attack evidence together to one belief equation. Here two independent evidences can be fused into a single belief function Z.
- The last module is the Decision Making module. We adapt Pignistic probability function here to classify attack behavior from network traffic data.

3.3 Fuzzy Clustering

3.3.1 General Formulation

Let's assume the available information in the training set from either USB or Wi-Fi contains *N* network traffic connections, and each of them is composed of *n* distinct features with positive numeric values. We denote the training set as *T*, the training connection as *x*,

$$T = \{x_1, x_2, x_3, \ldots, x_n\} \tag{3.1}$$

and the set of features in each connection as *F*.

$$F = \{f_1, f_2, f_3, \ldots, f_n\} \tag{3.2}$$

Data Clustering is trying to collect the same feature of attacks in one cluster, and obvious different with other features in other clusters. Fuzzy C-means clustering algorithm is the

soft partition of K-means clustering algorithm. Each membership is assigned to determine a certain degree of clustering which each data belongs to. Basically the main idea is to divide the n into c fuzzy groups and update the clustering center till reach some spectation.

The class set is *L* and it includes a number of *p* possible classes.

$$L = \{l_1, l_2, l_3, \ldots, l_n\} \tag{3.3}$$

Because a training connection sometimes could not be crisply defined as normality or abnormality in classification, we apply fuzzy c-Means clustering technique to deal with the above uncertainty. When the clustering operation is finished, we can obtain a set of cluster centers *C* and a membership partition matrix *U*. within each row of *U*, *i* is the connection number of the training set.

$$C = \{c_1, c_2, c_3, \ldots, c_p\} \tag{3.4}$$

$$U_{i\times p} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1p} \\ u_{21} & u_{22} & \cdots & u_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ u_{i1} & u_{i2} & \cdots & u_{ip} \end{bmatrix} \tag{3.5}$$

Within a vector (connection) of *U*, $U_{ij} \in [0,1]; \forall_i = 1,2,\ldots, p$ the degree of membership of vector x_i is in the cluster *j*. The membership grades are treated intuitively to be our degrees of confidence on classes that a connection can belong to. Consequently, we can build *p* decision rules from a connection and each one consists of a number of feature values *F*, a class label *l*, and a confidence value α.

$$R_U = \{r_U\} \quad where \quad r_U : \langle F_i, l_q \rangle, \ \alpha \tag{3.6}$$

In the training phase, the summation of the degrees of confidence on rules generated from a trainining data sets equals to 1.

$$\sum_{q=1}^{p} \alpha_{iq} = 1 \tag{3.7}$$

where *i* is the connection number and *j* is the class number. The confidence values are in proportion to the correspondent membership grades that connection belongs to certain classes.

In addition to the rules created from membership partition matrix *U*, a number of *p* rules are generated from the cluster centers. In each rule, the antecedent part includes *n* values of a cluster center and the corresponding class label. The degree of confidence is designated to 1 because we have full confidence that the cluster center should belong to that partitioned class without any doubt.

$$R_C = \{r_C\} \quad where \quad r_C : \langle c_q, l_q \rangle, \ \alpha = 1 \tag{3.8}$$

Using equations 3.6 and 3.8, we generate (*N+1*) X *p* rules in rule set *R*. The evidence is used to assign beliefs to the result of training data.

$$R = R_U \cup R_u \tag{3.9}$$

Equation 3.10 shows the function of Fuzzy C-Means algorithm. The gaol is to minimize the the function *j*.

$$J(U,C) = \sum_{i=1}^{N} \sum_{q=1}^{p} u_{iq}{}^{\beta} \left\| x_i - c_q \right\|^2 \tag{3.10}$$

- x_i is the i connection of the training set, c_q is the center of cluster *q*, and u_{iq} is the membership grade of x_i in the cluster *j* with a value between 0 and 1.

- $\| \ \|$ denotes norm expressing the distance or similarity between any measured data and the cluster center.
- $\|x_i - c_q\|$ represents the deviation of data x_i with c_q. And it shows the dissimilarity between x_i and c_q of in cluster *j*.
- The parameter β is a weighting exponent on fuzzy membership and $\forall \beta \in [1, \infty]$. β is the degree of fuzziness and it control the degree of membership grades. When β is 1 which is the minimum number, the Fuzzy C-Means is actually a hard c-Means algorithm. Normally, its value is between 1.25 to 2.

In order to get the minimum of *j*, we have the following equations 3.11&3.12. The membership grades u_{ij} and cluster centers c_j are calculated by the following expressions.

$$c_q = \frac{\sum_{i=1}^{N} u_{iq}{}^{\beta} x_i}{\sum_{i=1}^{N} u_{iq}{}^{\beta}} \qquad (3.11)$$

$$\forall q = 1,2,\cdots,P$$

$$u_{iq} = \frac{1}{\sum_{k=1}^{p} \left(\frac{\|x_i - c_q\|}{\|x_i - c_k\|} \right)^{\frac{2}{\beta-1}}} \qquad (3.12)$$

Fuzzy C-Means upgrade the cluster center to certain values by changing the cluster centers and membership grades. This upgrading stops when $\max_{iq} \left| u_{iq}^{k+1} - u_{iq}^{k} \right| < \varepsilon$ where ε is a selected threshold between 0 and 1, and *k* is the number of iterations.

Class with a higher membership grade belongs to the certain class (closer to the cluster center), and the class with a lower membership grade doesn't belong to the certain class (further to the cluster center). The fuzzy C mean algorithm is shown as below.

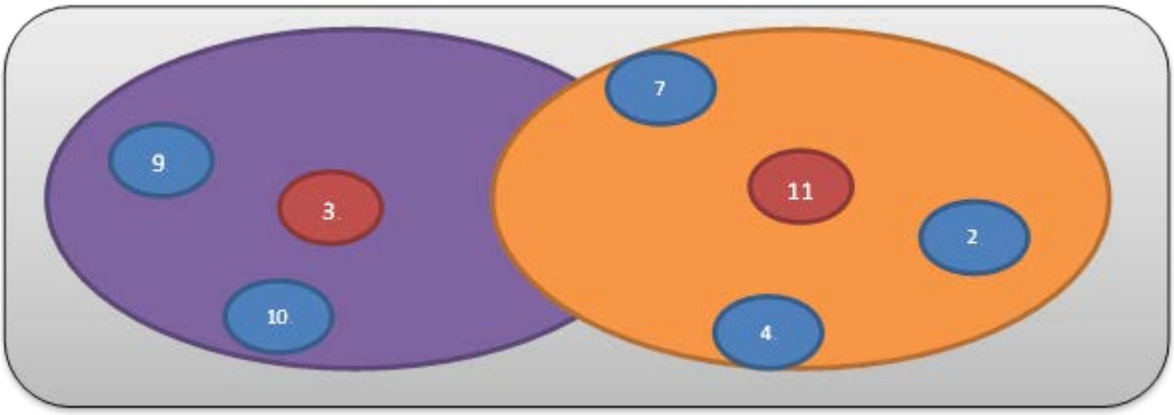

FIGURE 3.3 FCM EXAMPLE

Algorithm: Fuzzy C Means

Step1: Generate U and V

Step 2: At k-iteration, obtain centers vectors $C^k = [c_q]$

$$c_q = \frac{\sum_{i=1}^{N} u_{iq}{}^{\beta} x_i}{\sum_{i=1}^{N} u_{iq}{}^{\beta}}$$

$$\forall q = 1,2,\cdots,P$$

Step 3: Update $U^k and U^{k+1}$

$$u_{iq} = \frac{1}{\sum_{k=1}^{p} \left(\frac{\|x_i - c_q\|}{\|x_i - c_k\|} \right)^{\frac{2}{\beta-1}}}$$

Step 4: if then STOP;

Otherwise return to step 2

3.3.2 An Example

For example: There are the following node of 2, 3, 4, 7, 9, 10, and 11 shown in figure 3.3. And we have initial centroid 3 & 11. Here we let σ=2 in equation 3.12, and the following equation 3.13 should be held.

$$u_{iq} = \frac{1}{\sum_{k=1}^{p}\left(\frac{\|x_i - c_q\|}{\|x_i - c_k\|}\right)^2} \tag{3.13}$$

For node 2 which is the 1st element, we use the equation to calculate the membership of 1st node to 1st cluster which is:

$$U_{11} = \frac{1}{\left(\frac{2-3}{2-3}\right)^{\frac{2}{2-1}} + \left(\frac{2-3}{2-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+\frac{1}{81}} = \frac{81}{82} = 98.78\% \tag{3.14}$$

Then we have the membership of 1st node to 2nd cluster which is

$$U_{12} = \frac{1}{\left(\frac{2-11}{2-3}\right)^{\frac{2}{2-1}} + \left(\frac{2-11}{2-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+81} = \frac{1}{82} = 1.22\% \tag{3.15}$$

For node 3 which is the 2nd element, we use the equation again to calculate the membership of 2nd node to 1st cluster which is:

$$U_{21} = \frac{1}{\left(\frac{3-3}{3-3}\right)^{\frac{2}{2-1}} + \left(\frac{3-3}{3-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+0} = \frac{1}{1} = 100\% \tag{3.16}$$

Then the membership of 2nd t node to 2nd cluster which is $U_{22} = 0\%$

For node 4 which is the 3rd element, we use the equation again to calculate the membership of 3rd node to 1st cluster which is:

$$U_{31} = \frac{1}{\left(\frac{4-3}{4-3}\right)^{\frac{2}{2-1}} + \left(\frac{4-3}{4-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+\frac{1}{49}} = \frac{49}{50} = 98\% \quad (3.17)$$

Then we have the membership of 3rd node to 2nd cluster which is

$$U_{32} = \frac{1}{\left(\frac{4-11}{4-3}\right)^{\frac{2}{2-1}} + \left(\frac{4-11}{4-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+49} = \frac{1}{50} = 2\% \quad (3.18)$$

For node 7 which is the 4th element, we use the equation again to calculate the membership of 4th node to 1st cluster which is:

$$U_{41} = \frac{1}{\left(\frac{7-3}{7-3}\right)^{\frac{2}{2-1}} + \left(\frac{7-3}{7-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+1} = \frac{1}{2} = 50\% \quad (3.19)$$

Then we have the membership of 4th node to 2nd cluster which is

$$U_{42} = \frac{1}{\left(\frac{7-11}{7-3}\right)^{\frac{2}{2-1}} + \left(\frac{7-11}{7-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+1} = \frac{1}{2} = 50\% \quad (3.20)$$

For node 9 which is the 5th element, we use the equation again to calculate the membership of 5th node to 1st cluster which is:

$$U_{51} = \frac{1}{\left(\frac{9-3}{9-3}\right)^{\frac{2}{2-1}} + \left(\frac{9-3}{9-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+9} = \frac{1}{10} = 10\% \quad (3.21)$$

Then the membership of 5th node to 2nd cluster which is

$$U_{52} = \frac{1}{\left(\frac{9-11}{9-3}\right)^{\frac{2}{2-1}} + \left(\frac{9-11}{9-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+\frac{1}{9}} = \frac{9}{10} = 90\% \quad (3.22)$$

For node 10 which is the 6th element, we use the equation again to calculate the membership of 6th node to 1st cluster which is:

$$U_{61} = \frac{1}{\left(\frac{10-3}{10-3}\right)^{\frac{2}{2-1}} + \left(\frac{10-3}{10-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+49} = \frac{1}{50} = 2\% \tag{3.23}$$

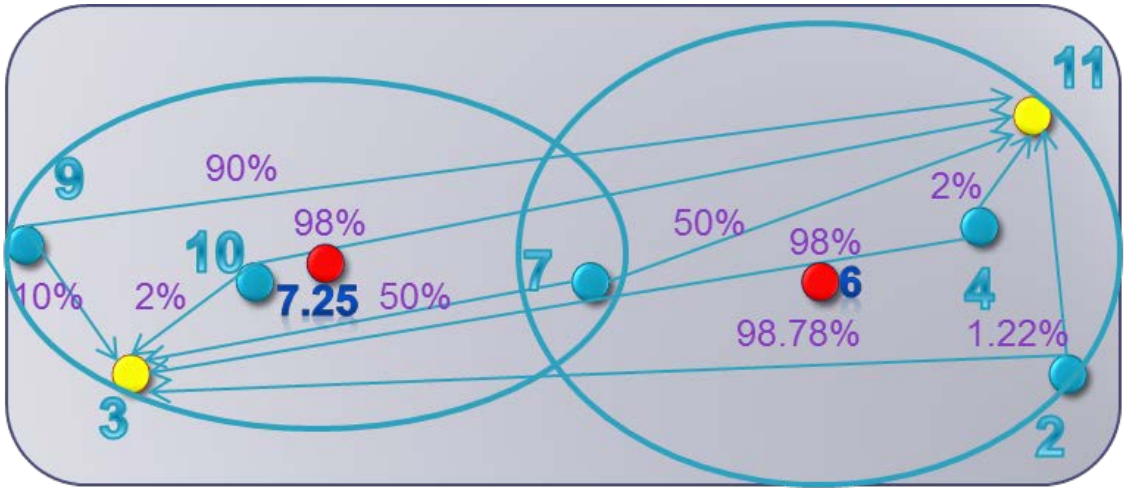

FIGURE 3.4 NODES WITH DISTANCE TO THE CENTROID AND NEW CENTROIDS

Then the membership of 6th node to 2nd cluster which is

$$U_{62} = \frac{1}{\left(\frac{10-11}{10-3}\right)^{\frac{2}{2-1}} + \left(\frac{10-11}{10-11}\right)^{\frac{2}{2-1}}} = \frac{1}{1+\frac{1}{49}} = \frac{49}{50} = 98\% \tag{3.24}$$

For node 11 which is the 7th element, we use the equation again to calculate the membership of 7th node to 1st cluster which is:

$U_{71} = 0\%$

Then the membership of 7th node to 2nd cluster which is:

$U_{72} = 100\%$

Then we have all the nodes with distance to the centroid node 3 and 11 in the figure 3.4. We update the new Centroids: new centroid: 7.25 and new centroid: 6.

3.4 Dempster-Shafer Theory

The evidence of incoming traffic is unavaible for attacks or normal behaviors. Also, the training set data is not efficient enough to provide accurate information. Thus, the D-S theory is used into this classification phase to solve the limited information for training data. We don't need to classify the traffic based on the previous consumption of the training data sets. The probability is calculated through this theory. Usually the attacks are hiding in the traffic data and are not easy to find out. This D-S theory can make decisions on data whether its normal or attack with limited information provided. In this phase, the pieces of evidences will be derived from the decision rules of the training phase. Here, we apply k-NN rule to find the most informative *k* nearest training connections of *v*. By using these *k* connections, we then can find the corresponding decision rules. Also, we use weighted k-NN rule to assign different weights *w* to these rules in order to differentiate the degrees of importance.

3.4.1 K Nearest Neighbor Rule

Now let's assume *v* be an incoming connection to be classified. In order to classify it into the correct class, D-S theory is used to measure and combine pieces of evidence derived from the set of decision rules. This theory starts by defining the set of class labels *L* as the *frame* of the problem domain. The possible subset *A* of *L* represent hypothesis that one could present evidence. To classify *v* means to assign it to one of members in *L*.

For classifying *v* into the correct class, we treat the set of decision rules as pieces of evidence that alters our degrees of belief on which class *v* should belong to. If the distance is large between *v* and a decision rule, it implies that the rule only has a little influence on *v*. On the other hand, we have stronger belief that *v* should belong to the same class of the rule if *v* is "close" to it, which means the distance has a smaller value.

$$w_t = \begin{cases} \dfrac{d(x_K, x_v) - d(x_t, x_v)}{d(x_K, x_v) - d(x_1, x_v)} & d(x_K, x_v) \neq d(x_1, x_v) \\ 1 & d(x_K, x_v) = d(x_1, x_v) \end{cases} \quad \forall \mathrm{t} \in (1,....\mathrm{K}) \tag{3.25}$$

All the distances are calculated from v to all rules. x_i is the i^{th} rule, x_k is the farthest rule, x_1 is the nearest rule of *v*, *d* is the Euclidean distance from *v* and to decision rule. The influence is assigned to each v. The factor is bigger means the rule is closer to v, and the factor is smaller means the rule is further to v. We try to find out the nearest neighbor of v with the weight value of 1. And the furthest neighbor of v with weight value of 0. So this factor value is between 0 and 1.

3.4.2 Dempster Shafer Theory

Dempster-Shafer theory describes the sample space as evidence or belief functions. We use class labels L as the length of the sample space domain. The subset *A* of *L* is the hypothesis supporting evidence. So A and null set ϕ is a power set and denoted as 2^L. Let *v* be the classification traffic data. We try to classify v to one of the *p* classes in levels L. $v \in l_q$, q = 1, 2, …, p. We use the mass function denoted as $m(\cdot)$ as mapping function for m: $2^L \rightarrow$ [0, 1], so we have

$$\sum_{A \subseteq L} m(A) = 1 \tag{3.26}$$

$$m(\phi) = 0 \tag{3.27}$$

where $A \subseteq L$ is a *focal element* of m if $m(A) > 0$. The quantity $m(A)$ is probability which is a portion of evidences.

By adapting Dempster-Shafer theory, the degree of belief is quantified by *mass function* which is denoted as m.

$$m(l_q) = w \cdot \alpha \tag{3.28}$$

where q is the class number. Up to this stage, each rule creates a belief assignment indicating the degree that v belongs to a certain class. Nevertheless, we need to notice that a belief should also be designated to the frame (with every class labels). The reason is that only part of our beliefs is committed to single classes for a given training connection, and the rest of our belief should be assigned to the whole class set. According to Dempster-Shafer theory, the summation of all mass functions inferred from one training connection is equal to 1. Thus, the belief belonged to the frame becomes one minus the summation of beliefs of all single classes.

$$m(L) = 1 - \sum_{i=1}^{p} m_i(l_q) \tag{3.29}$$

Generally speaking, the mass function is a piece of evidence that supports certain hypothesis concerning to the class member of a rule. When more evidences appear with same class label, these evidences can be integrated to generate a single belief function which represents the total support for the same class. Now assume that there are two mass functions m_1 and m_2 induced by distinct items of evidences X and Y. By using *Dempster Rule of Combination*, these two independent evidences can be fused into a single belief

function that expresses the support of the hypotheses in both evidences. The combination result is called *orthogonal sum* of m_1 and m_2 and noted as $m = m_1 \oplus m_2$.

$$m(Z) = \frac{\sum_{X \cap Y = Z} m_1(X) \cdot m_2(Y)}{\sum_{X \cap Y \neq \varnothing} m_1(X) \cdot m_2(Y)} = \frac{\sum_{X \cap Y = Z} m_1(X) \cdot m_2(Y)}{1 - \sum_{X \cap Y = \varnothing} m_1(X) \cdot m_2(Y)} \tag{3.30}$$

Based on the equation 3.28, the belief function *Bel* and plausibility function *Pl* are used to show hypotheses. We have

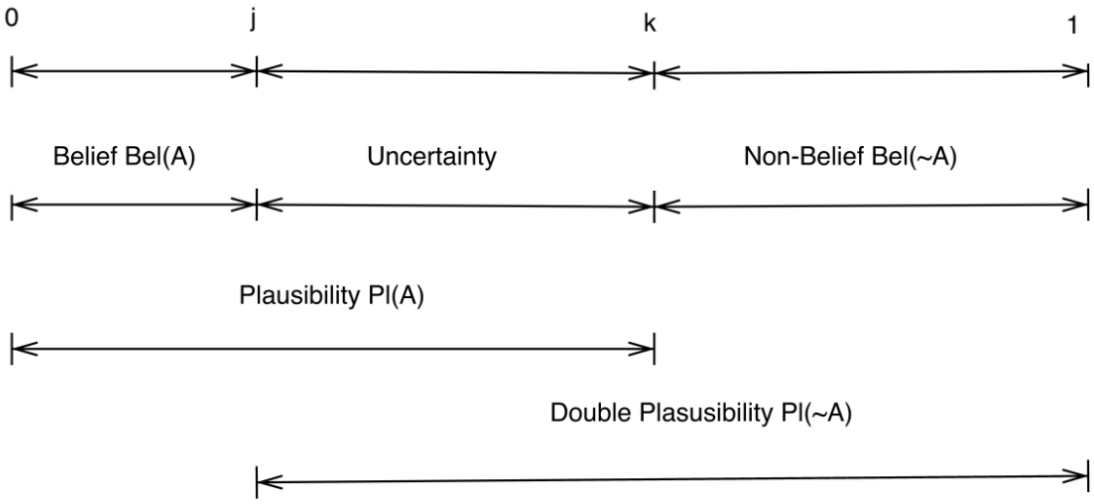

FIGURE 3.5 FUNCTION OF BELIEF AND PLAUSIBILITY

$$Bel(l_q) = m(l_q) \tag{3.30}$$

and

$$pl(l_q) = 1 - bel(\bar{l}_q) \tag{3.31}$$

where q is class number and $\bar{l}_q$ is the hypothesis. Belief function can be regarded as a lower bound of the evidence. Plausibility is an upper bound on the belief. We can see the bel is the lower boundary and the pl is the upper boundary of hypothesis. The gap between is the uncertainty part of evidence in Figure 3.5.

Function of Believe and Plausibility

Two mass functions m_1 and m_2 can be fused into a single belief function. The combination result is noted as $m = m_1 \oplus m_2$

$$m(Z) = \frac{\sum_{X \cap Y - Z} m_1(X) \cdot m_2(Y)}{\sum_{X \cap Y \neq \phi} m_1(X) \cdot m_2(Y)} = \left(\sum_{X \cap Y - Z} m_1(X) \cdot m_2(Y) \right) \cdot k^{-1} \tag{3.32}$$

Where

$$k^{-1} = \left(\sum_{X \cap Y \neq \phi} m_1(X) \cdot m_2(Y) \right)^{-1} = \left(1 - \sum_{X \cap Y \neq \phi} m_1(X) \cdot m_2(Y) \right)^{-1} \tag{3.33}$$

The factor k^{-1} is the renormalization constant. We use the equation 3.32 to generate the final belief of a single class. The two belief functions *Bel(N)* and *Bel(A)* can be generated through the above evidence.

3.4.3 Decision Making

After having all the *fused mass functions*, the final decision is made by introducing the *pignistic probability function*. It is illustrated as follows:

$$Bp(l_q) = m(l_q) + \frac{m(L)}{p} \tag{3.34}$$

where q is the class number and p is the number of classes. The function quantifies our beliefs to individual classes with pignistic probability distribution. For making an optimal decision, v is assigned to a class with the highest pignistic probability. Hence, the incoming patterns of information can be classified as intrusion or non-intrusion activities.

3.4.4 An Example

The mass functions is 0.15 for normal class, and the mass function is 0.2 for attack class. According to equations 3.30 and 3.31, the belief for normal class is 0.15, and the plausibility for normal class is 0.8. For the attack class, the belief is 0.2 and the plausibility is 0.85.

Let $m_1(N) = 0.15$ and $m_1(A) = 0.2$

According to equations $Bel(l_q) = m(l_q)$ and $pl(l_q) = 1 - bel(\bar{l}_q)$, we calculate the belief and plausibility.

$$Bel(N) = m_1(N) = 0.15 \tag{3.35}$$

$$Bel(A) = m_1(A) = 0.2 \tag{3.36}$$

$$Pl(N) = 1 - Bel(\overline{N}) = 1 - Bel(A) = 1 - 0.2 = 0.8 \tag{3.37}$$

$$Pl(A) = 1 - Bel(\overline{A}) = 1 - Bel(N) = 1 - 0.15 = 0.85 \tag{3.38}$$

$$Un(N) = Bel(N) - Pl(N) = 0.80 - 015 = 0.65 \tag{3.39}$$

$$Un(A) = Bel(A) - Pl(A) = 0.85 - 0.20 = 0.65 \tag{3.40}$$

The *frame* is

$$L = \{l_1, l_2\} = \{N, A\} \tag{3.41}$$

Then we add two more pieces of evidence mass functions are 0.25 and 0.7 for normal class and attack class, respectively. Let $m_2(N) = 0.25$ and $m_2(A) = 0.7$

Table 3.1 shows the Connection where normal is denoted by N and attack is denoted by A. The gap between belief and plausibility is denoted as U.

First, the factor k^{-1} is calculated. Then the fused mass functions can be calculated by using Equation 3.43. Equations 3.44 to 3.46 are the final results.

$$
\begin{aligned}
k^{-1} &= \left(\sum_{N \cap A \neq \phi} m_1(N) \cdot m_2(A) \right)^{-1} \\
&= \left(1 - \sum_{N \cap A \neq \phi} m_1(N) \cdot m_2(A) \right)^{-1} \\
&= (1 - [m_1(N) \cap m_2(A) + m_1(A) \cap m_2(N)]^{-1} \\
&= (1 - (0.11 + 0.55))^{-1} \\
&= (1 - 0.06)^{-1} \\
&= (0.84)^{-1} \\
&= 1.19
\end{aligned}
\tag{3.42}
$$

$$
m(l_q) = \left[m_1(l_q) \cdot m_2(l_q) + m_1(l_q) m_2(L) + m_1(L) \cdot m_2(l_q) \right] \cdot k^{-1} \tag{3.43}
$$

$$
\begin{aligned}
m(N) &= m_1 \oplus m_2(N) \\
&= \left[m_1(N) \cdot m_2(N) + m_1(N) \cdot m_2(N, A) + m_1(N, A) \cdot m_2(N) \right] \cdot k^{-1} \\
&(0.04 + 0.01 + 0.16) \cdot k^{-1} \\
&= 0.21 \times 1.19 \\
&= 0.25 \\
&= Bel(N)
\end{aligned}
\tag{3.44}
$$

$$
\begin{aligned}
m(A) &= m_1 \oplus m_2(A) \\
&= \left[m_1(A) \cdot m_2(A) + m_1(A) \cdot m_2(N, A) + m_1(N, A) \cdot m_2(A) \right] \cdot k^{-1} \\
&(0.14 + 0.01 + 0.46) \cdot k^{-1} \\
&= 0.61 \times 1.19 \\
&= 0.73 \\
&= Bel(A)
\end{aligned}
\tag{3.45}
$$

$$
\begin{aligned}
m(L) = m(N, A) &= m_1 \oplus m_2(N, A) \\
&= \left[m_1(N, A) \cdot m_2(N, A) \right] \cdot k^{-1} \\
&= 0.03 \cdot k^{-1} \\
&= 0.04
\end{aligned}
\tag{3.46}
$$

The plausibility and gap between belief and plausibility for both normal and attack classes can be derived using Equations 3.47 to 3.52.

$$Bel(N) = m_2(N) = 0.25 \tag{3.47}$$

$$Bel(A) = m_2(A) = 0.73 \tag{3.48}$$

$$\begin{aligned} Pl(N) &= 1 - Bel(\overline{N}) \\ &= 1 - Bel(A) \\ &= 1 - 0.73 \\ &= 0.27 \end{aligned} \tag{3.49}$$

$$\begin{aligned} Pl(A) &= 1 - Bel(\overline{A}) \\ &= 1 - Bel(N) \\ &= 1 - 0.25 \\ &= 0.75 \end{aligned} \tag{3.50}$$

$$\begin{aligned} Un(N) &= Bel(N) - Pl(N) \\ &= 0.27 - 0.25 \\ &= 0.02 \end{aligned} \tag{3.51}$$

$$\begin{aligned} Un(A) &= Bel(A) - Pl(A) \\ &= 0.75 - 0.73 \\ &= 0.02 \end{aligned} \tag{3.52}$$

The gap between belief and plausibility is 0.08. Now we can see when we add two more evidence to it, the gap reduced from 0.65 to 0.08, which shows that this connection should be attack.

The degrees of final belief on normal and attack classes are shown in equation 3.53&3.54. Table 3.2 is the final result.

$$Bp(N) = m(N) + \frac{m(L)}{2} = 0.25 + \frac{0.04}{2} = 0.27 \tag{3.53}$$

$$Bp(A) = m(A) + \frac{m(L)}{2} = 0.73 + \frac{0.04}{2} = 0.75 \tag{3.54}$$

Table 3.1 CONNECTION

	$m_1(N) = 0.15$	$m_1(A) = 0.2$	$m_1(N,A) = 0.65$
$m_2(N) = 0.25$	$m(N) = 0.04$	$m_1(N \cap A) = 0.05$	$m(N) = 0.16$
$m_2(A) = 0.7$	$m_1(N \cap A) = 0.11$	$m(A) = 0.14$	$m(A) = 0.46$
$m_2(N,A) = 0.05$	$m(N) = 0.01$	$m(A) = 0.01$	$m_1(N \cap A) = 0.03$

Table 3.2 RESULTS

	{N}	**{A}**	**{N,A}**
m_1	0.15	0.2	0.65
Bel_1	0.15	0.2	1
Pl_1	0.8	0.85	1
m_2	0.25	0.7	0.05
Bel_2	0.25	0.7	1
Pl_2	0.3	0.75	1
m	0.25	0.73	0.04
Bel	0.25	0.73	1
Pl	0.27	0.75	1
BP	0.02	0.02	
U	0.27	0.75	

CHAPTER 4

EVALUATION OF FUZZY BELIEF K-NN SINGLE CLASSIFIER

In this chapter, we introduce the DARPA KDD 99 data sets as our benchmark in our experiment.

4.1 DARPA KDD99 Data Set

It is very hard to generate large amount of computer networking data to test our classifier. Therefore, we use the existing data set to train and test our system. In order to compare with other researchers result, we choose not to simulate the data by ourselves. As we know it needs a lot of work to simulate the attacks within a network environment. DARPA KDD99 Intrusion Detection Evaluation data are used in my dissertation as a benchmark for analyzing the performance of the proposed classifiers. It was first generated by MIT lab to monitor the network traffic for two weeks. It has been used by many researchers to test their IDS and its open to public with large number of network traffic activities including attacks and normal. There are three sets in KDD99 data: whole KDD, 10% KDD, and corrected KDD. In this dissertation, the 10% KDD data is used as training set, and the corrected KDD data is used as testing set. The patterns the classification use to detect attacks are basically the features of the data. We can select less correlated feature subsets to strengthen the patterns distinguishing in the machine learning process. Mapping is also an important step to link the input data to the corresponding features.

TABLE 4.1 CONNECTION DISTRIBUTION

Data Set	Normal	DoS	R2L	U2R	Probe	Total
Training Set	97,277	391,458	1,126	52	4,107	494,020
Testing Set	60,593	229,853	16,189	228	4,166	311,029

TABLE 4.2 THIRTY NINE ATTACKS

DoS	R2L	U2R	Probe
apache2, back,	ftp_write,guess_passwd,	buffer_overflow,	ipsweep, mscan,
land, mailbomb,	imap, multihop, named,	httptunnel,	nmap, portsweep,
netpune, pod,	phf, sendmail,	loadmodule, perl,	saint, satan.
processtable,	snmpgetattack,	ps, rootkit,	
smurf, teardrop,	snmpguess, spy,	sqlattack, xterm.	
udpstorm.	warezclient, warezmaster,		
	worm, xlock, xsnoop.		

4.2 Data Sets Selection

The *KDD99* data set that is made up of a large number of network traffic connections. Each connection is represented with 41 features plus a label of either normal or a type of attack. Totally 39 attack types are included and are fall into four main classes, *DoS*, *Probe*, *U2R*, and *R2L*. There are 22 types of attacks in training set and 17 types of attacks in testing sets. Table 4.1 shows the training set and testing set connection respectively. Table 4.2 shows the 22 types of attacks in training set are marked underline. Figure 4.1 shows the distributions of *DoS*, *Probe*, *U2R*, and *R2L* attacks. As we can see here the DoS and Probe attacks are very common in computer networking traffic connections, while U2R and R2L attacks are very rare in computer networking traffic connections. The distribution of these four attacks are not even in Darpar KDD 99 data sets.

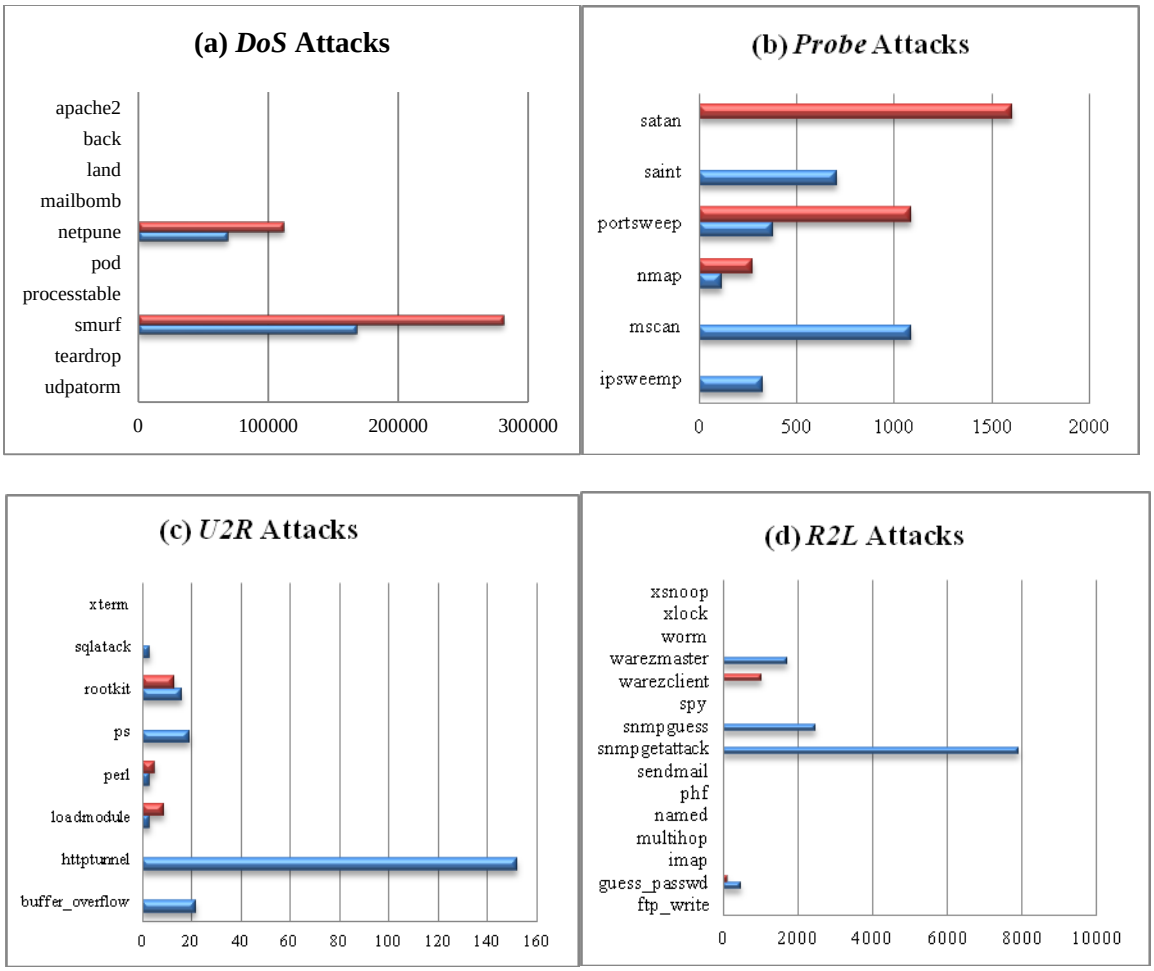

FIGURE 4.1 DISTRIBUTIONS OF FOUR KDD99 ATTACK CATEGORIES:
: Training Sets : Testing Sets

4.3 Data Sets Preprocessing

The fuzzy belief experiments are performed on the binary (normal/attack) classification. Our two-stage classifier and ensemble classifier experiments are performed on multi-label classification. To minimize the inaccuracy and variation factor of experiment results, 10 trials are performed in every detection task. In each trial, only a very small amount of connections are randomly selected from reduced training and testing sets. It not only

speeds up the classification process but also simulates the uncertainty caused by lack of network traffic information.

For the binary classification, the four training sets have 545 *DoS* attacks, 213 *Probe* attacks, 52 *U2R* attacks, and 99 *R2L* attacks, respectively and each set has a number of 878 normal connections. The four testing sets have 235 *DoS* attacks, 268 *Probe* attacks, 215 *U2R* attacks, and 291 *R2L* attacks, respectively and each set has 479 normal connections. The sizes of the original training and testing sets are reduced by removing the duplicated connections in Table 4.3.

4.4 Features in Data Sets

The set describes each connection in terms of 41 features plus a label of either normal or a type of attack as shown in table 4.4. The features is the same with the attributes to describe each connections in the networking traffic. The content of these features are continuous, discrete, or symbolic with vary scales and ranges. These features can be classified into four classes, *basic, content, time-based,* and *host-based* features. Features 1 to 9 are *basic features* that are derived from packet header without inspecting the payload. Features 10 to 22 are *content features* that are obtained by analyzing the payload of the original TCP packets. Features 23 to 31 are *time-based traffic features* that capture properties of connections in the past 2 seconds. Features 32 to 41 are *host-based traffic features* that examine a number of connections using a window of 100 connections instead of a 2-second time window. For example, the protocol_type feature shows the traffic connection belongs to Ethernet, local talk, token ring, FDDI, or ATM. The duration feature shows the period that the connection last.

TABLE 4.3 REDUCED TRAINING AND TESTING SETS

	Total	Normal	DoS	Probe	U2R	R2L
Training Sets	1787	878	545	213	52	99
Testing Sets	1488	479	235	268	215	291

TABLE 4.4 41 FEATURES IN KDD 99 DATA SETS

Feature# + Feature Name	**Feature# + Feature Name**
1 duration	**22 is guest login**
2 protocol type	**23 count**
3 service	**24 serror rate**
4 src byte	**25 rerror rate**
5 dst byte	**26 same srv rate**
6 flag	**27 diff srv rate**
7 land	**28 srv count**
8 wrongfragment	**29 srv serror rate**
9 urgent	**30 srv rerror rate**
10 hot	**31 srv diff host rate**
11 num failed logins	**32 dst host count**
12 logged in	**33 dst host srv count**
13 num compromised	**34 dst host same srv count**
14 root shell	**35 dst host diff srv count**
15 su attempted	**36 dst host same src port rate**
16 num root	**37 dst host srv diff host rate**
17 num file creations	**38 dst host serror rate**
18 num shells	**39 dst host srv serror rate**
19 num access shells	**40 dst host rerror rate**
20 num outbound cmds	**41 dst host srv rerror rate**

4.5 Experimental Result Expression

For detecting the attacks, training and testing are performed in each trial. In the training phase, our proposed method and other methods are constructed using the limited and ambiguous training data. The testing data are then fed into the trained classifier to identify intrusions in the testing phase. We evaluate the performances using false positive rate (FPR) and detection rate (DR), and Receiver Operating Characteristics (ROC) graphs. The false positive rate is the percentage of normal connections that are incorrectly identified as attacks. The detection rate is the percentage of attacks that are correctly identified. The ROC graphs that plot FPRs on the *X* axis and DRs on the *Y* axis. In the tasks of detecting attacks, the differences in both FPRs and DRs are very slight for different kinds of classifiers. Since *DoS* and *Probe* attacks usually have frequent sequential patterns that are different from the normal connections, they can be easily separated from normal activities and thus all of three classifiers can achieve low FPRs and high DRs. On the contrary, *U2R* and *R2L* attacks do not have any intrusion only frequent sequential patterns. They are embedded in the data portions of the packets and normally involve only a single connection. Therefore, most machine learning approaches would fail to achieve high DRs in *U2R* and *R2L* attacks. Although some classifiers have very low FPRs, it is because they treat most network traffic data as normal connections either they are normal or malicious activities.

CHAPTER 5

TWO-STAGE FUZZY KNN-DST CLASSIFIER FOR UNKNOWN ATTACKS

In this chapter, an innovative fuzzy classifier is proposed for effectively detecting both unknown attacks and known attacks with insufficient or inaccurate training information. The motivation for two-stage fuzzy classifier is introduced in section 5.1. Section 5.2, describes the proposed two-stage fuzzy KNN-DST IDS in detail. Firstly, a fuzzy C-means (FCM) algorithm is employed to softly compute and optimize clustering centers of the training datasets with some degree of fuzziness counting for inaccuracy and ambiguity in the training data. Subsequently, a distance-weighted k-NN (k nearest neighbors) classifier, combined with the Dempster-Shafer Theory (DST), is introduced to assess the belief functions and pignistic probabilities of the incoming data associated with each of known classes. Finally a two-stage intrusion detection scheme is implemented based on the obtained pignistic probabilities to determine if the input data are normal, one of the known attacks or an unknown attack. At second stage both neutral zone and entropy function are applied to detect unknown attacks. The experimental results show in section 5.3 that the new algorithm with entropy function outperforms algorithm with neutral zone and other intrusion detection algorithms and is especially effective in detecting unknown attacks. Section 5.4 draws some conclusions on our proposed method of two-stage classifier for detection unknown attacks.

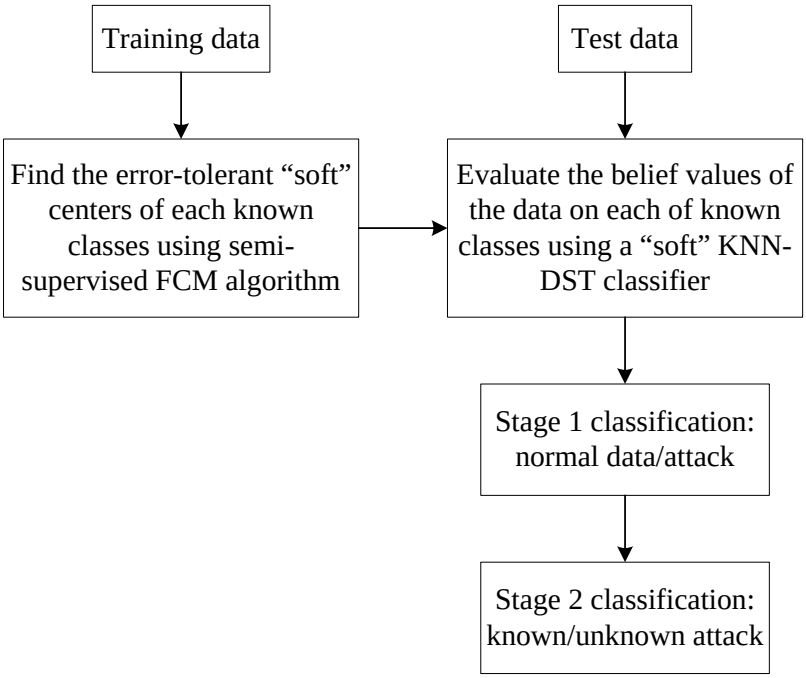

FIGURE 5.1 TWO-STAGE CLASSIFYING SCHEMES FOR UNKNOWN ATTACK DETECTION

5.1 Introduction

Basically, there are mainly two types of IDS: anomaly intrusion detection and misuse or signature-based intrusion detection. Anomaly intrusion detection tries to identify if the data traffic pattern is abnormal by comparing it with previously obtained normal traffic profiles; while the misuse methods detect intrusions by matching the data signature or feature vector to that of one of the known attacks. Although theoretically capable to detect unknown attacks, anomaly IDS are generally inefficient, time-consuming and very difficult to implement with poor performance due to the lack of training data. On the other hand, misuse IDS are very efficient in data classification, but they can only detect known attacks and suffer from high error detection rate especially when the attacks are unknown or the classifier is not properly or sufficiently trained. Most of current IDS algorithms are based on modern classification methods.

The current classifying algorithms used in IDS are typically designed based on the naïve Bayesian method [6,7], support vector machine [8-11], particle swarm optimization [12,13], generic algorithm [14], neural networks [15-18], k-nearest neighbor (KNN) methods [19,20], fuzzy c-means (FCM) methods [21,22], Dempster-Shafer theory of evidence [23-26] or other decision-tree based ad-hoc methods [27]. However, the detection performance of the current IDS are generally sensitive to the mismatching between the training and test data; and they could perform very poorly in the case of slight deviation of intrusive data pattern from known patterns or of an unknown attack. It is well-known that the forms of cyber attacks and internet hacking tactics are constantly changing and evolving and new internet "viruses" are created almost every day. Therefore, it is imperative that the performance of IDS is not markedly degraded when a known intrusion is morphed into a different form of attack or a new intrusion has a completely different profile. In this work, we will introduce an innovative two-stage fuzzy classifier embedded with "soft" and error-tolerant classification mechanism for effective detection of various malicious intrusions including unknown attacks. Fuzzy classifiers are known for tolerating training or test data errors or variations due to "soft" clustering and classification techniques involved, but none of the current fuzzy classifiers is capable to effectively detect both known and unknown attacks simultaneously. In addition, with this proposed IDS, the Dempster-Shafer Theory (DST) [28] is seamlessly combined with a distance-weight k-NN algorithm by fusing multiple "soft" independent evidences in order to assess the belief value of the input data belonging to each of the known classes. The two-stage classification in the algorithm is set firstly to determine whether the incoming traffic is normal or an attack and subsequently to determine

whether it is an unknown attack if the first stage detection is positive as abnormal connections.

5.2 A New Fuzzy KNN-DST Classifier for Unknown Intrusion Detection

The framework of the proposed fuzzy KNN-DST classifier is shown in Figure 5.1. Since the new classifier should be capable to detect unknown intrusions and mutated versions of known intrusions, we choose to use the k-nearest neighbor (k-NN) algorithm for its robust performance and its tolerance for inaccuracy and random errors in the input data.

In our developed new algorithm, a belief value of a test connection associated with a known class is softly measured based on the distance between the input data and the centroid of the class, and the Dempster-Shafter Theory (DST) is incorporated into the framework to fuse multiple evidences generated from a weighted k-NN algorithm to form a pignistic probability of a test connection belonging to a known class. The centers of known classes are softly defined and computed using a semi-supervised fuzzy c-means (FCM) algorithm from the training data.

Stage one classification in Figure 5.1 determines if a connection is normal data or an intrusion. If it is an abnormal intrusion, stage-two classification is needed to determine if it is one of known attacks or unknown attack. The details of the new classifier are given in the next two subsections.

5.2.1 Semi-Supervised Fuzzy C-Means Learning Algorithm

Let us assume the training set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_N\}$ contains N network traffic connections, and each of them is either normal connection or known attack. Each connection is represented by a distinct feature vector with positive numeric values.

Normally for computer network connections, the extracted feature vector consists of the source and destination bytes, the connection type, or the duration of a connection, and etc. The set of features generated from all data connections are assumed to be:

$$\mathbf{F} = \{\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3, ..., \mathbf{f}_N\} \tag{5.1}$$

We denote the set $\mathbf{L} = \{l_1, l_2, ..., l_P\}$ as P possible data classes, which include known attacks and the normal data stream. To avoid the crisp definition of a connection belonging to one of the classes, we employ the FCM algorithm allowing one traffic connection to belong to more than one class/cluster with varying membership values.

Firstly, we will try to divide the N traffic connections into P clusters/classes and each cluster is represented by its centroid, which is an element of $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_P\}$. In addition, a membership partition matrix $\boldsymbol{U}$ of size ($N \times P$) is used to measure the closeness of a data connection to each of the class centers. The membership matrix elements are defined by:

$$u_{iq} = \frac{\| \mathbf{f}_i, \mathbf{c}_q \|^{\frac{-2}{\beta-1}}}{\sum_{q=1}^{P} \| \mathbf{f}_i, \mathbf{c}_q \|^{\frac{-2}{\beta-1}}}, \; 1 \le i \le N, \; 1 \le q \le P \tag{5.2}$$

where u_{iq} of a value between 0 and 1 is the membership grade of the input data connection i in the cluster q, β ($\beta > 1$) is the weighting exponent representing the degree of the fuzziness for the membership grades, and $\| \mathbf{f}_i, \mathbf{c}_q \|$ represents the Mahalanobis distance between the data feature vector $\mathbf{f}_i$ and the centroid $\mathbf{c}_q$ of cluster q and is defined as:

$$\| \mathbf{f}_i, \mathbf{c}_q \| = \sqrt{(\mathbf{f}_i - \mathbf{c}_q)^T \boldsymbol{\Sigma}_q^{-1} (\mathbf{f}_i - \mathbf{c}_q)} \tag{5.3}$$

where $\mathbf{\Sigma}_q$ is the covariance matrix of the centroid vector of cluster *q*. Equation 5.3 becomes the Euclidean distance when $\mathbf{\Sigma}_q$ is the unity matrix.

The centroid of cluster *q* is further defined as:

$$\mathbf{c}_q = \frac{\sum_{i=1}^{N} u_{iq}{}^{\beta}\mathbf{f}_i}{\sum_{i=1}^{N} u_{iq}{}^{\beta}} \quad \forall q = 1,2,\cdots,P \tag{5.4}$$

The cluster centroids are iteratively optimized by minimizing the following dissimilarity function ***J(U,C)***:

$$J(U,C) = \sum_{i=1}^{N}\sum_{q=1}^{P} u_{iq}{}^{\beta} \,\|\, \mathbf{f}_i, \mathbf{c}_q \,\|^2$$
$$\text{Subject to}: \ \sum_{q=1}^{P} u_{iq} = 1, \forall i \tag{5.5}$$

With the FCM algorithm, we keep on upgrading $\mathbf{c}_q$ and u_{iq} iteratively until the dissimilarity function ***J(U,C)*** is minimized. The optimal cluster centroids $\mathbf{c}_q$ for the fuzzy classifier are found when the iteration stops with $\max_{i,q}\left|u_{iq}^{(\eta+1)} - u_{iq}^{(\eta)}\right| < \varepsilon$, where ε is a pre-selected threshold between 0 and 1, and η is the number of iterations. The initial values of the cluster centroids in (5.2) are obtained from the labelled training data directly. Therefore, the iterations in (5.2)-(5.5) normally can converge quickly. Since the class information of the labelled training data is used in the FCM algorithm, the learning process is considered to be semi-supervised.

5.2.2 A Two-Stage KNN-DST Classifier

With the centroids of the known clusters found through the Fuzzy C-means algorithm, we try to employ a weighted k-NN approach in our classifier by considering the k-nearest neighbors of a new test data connection $\mathbf{x}_v$ in the training dataset. Let us associate the test data connection $\mathbf{x}_v$ with the class l_q of one of the k-nearest neighbors $\mathbf{f}_q$ by defining a fuzzy membership function u_{vq} based on the distance between the test data and the class centroid $\boldsymbol{c}_q$ that is similar to (5.2).

However, the association between the test data and class l_q also should be affected by the distance between the test data $\mathbf{x}_v$ and the training neighbor $\mathbf{f}_q$. If there is a large distance between $\mathbf{x}_v$ and one of the k-nearest training records, the probability of $\mathbf{x}_v$ and the training record belonging to the same class is small. Therefore, the membership grade of a test data record belonging to a class should be weighted based on the distance between the test data and its nearest neighbors in the k-NN algorithm.

We assume that the K nearest neighboring training data records of $\mathbf{x}_v$ are represented by the set of their feature spaces $\mathbf{F}_K = \{\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_K\}$ and $\{\|\mathbf{f}_v, \mathbf{f}_1\|, \|\mathbf{f}_v, \mathbf{f}_2\|, \cdots, \|\mathbf{f}_v, \mathbf{f}_K\|\}$ is the set of the corresponding distances between the test data feature $\mathbf{f}_v$ and the K nearest training samples $\mathbf{f}_k$ in ascending order. Hence, the membership grade of a test data record belonging to a class in the weighted k-NN algorithm should be weighted with the following coefficient.

$$w_\gamma = \begin{cases} \dfrac{\|\mathbf{f}_K, \mathbf{f}_v\| - \|\mathbf{f}_r, \mathbf{f}_v\|}{\|\mathbf{f}_K, \mathbf{f}_v\| - \|\mathbf{f}_1, \mathbf{f}_v\|}, & \|\mathbf{f}_K, \mathbf{f}_v\| \neq \|\mathbf{f}_1, \mathbf{f}_v\|,\ 1 \le \gamma \le K \\ 1, & \|\mathbf{f}_K, \mathbf{f}_v\| = \|\mathbf{f}_1, \mathbf{f}_v\| \end{cases} \tag{5.6}$$

where $\mathbf{f}_1$ is the first nearest neighbor of $\mathbf{f}_v$ and $\mathbf{f}_k$ is the k^{th} nearest neighbor of f_v and the weight w_γ is assigned to modify the association between the connection $\mathbf{f}_v$ and the class of its γ^{th} nearest neighbor. The closer the neighbors are, the greater weights they are assigned to. In the weighted k-NN algorithm, each of the k nearest neighbors and its class designation contribute to the classification by providing an independent piece of evidence. Since DST is an evidence theory that can be used to combine separate pieces of evidence to determine the probability of an incident [24], we will use the DST in our classifier to fuse the class information obtained from all k-nearest neighbors to facilitate the intrusion detection work. The goal is to try to classify the new connection $\mathbf{f}_v$ into one of the members in class label set $L = \{1, 2, \cdots, P\}$.

DST describes the probability of a test data sample belonging to a class using belief functions and the degree of belief is quantified by a *mass function* denoted as *m.* The term $m_\gamma(l_q)$ of $\mathbf{f}_\gamma$ can be treated as a piece of evidence that contributes to our belief that $\mathbf{f}_v$ belongs to class l_q. Since only a part of our belief is committed to l_q and represented by $m_\gamma(l_q)$, the rest of the belief is assigned to the whole frame of discernment represented by $m_\gamma(L)$. Specifically, the belief functions of the input data connection $\mathbf{f}_v$ belonging to class l_q and the whole frame L due to the evidence from one of its neighbor $\mathbf{f}_\gamma$ are defined, respectively, as:

$$m_\gamma(l_q) = \zeta \cdot w_\gamma \cdot u_{vq} = \zeta \cdot w_\gamma \cdot \frac{\| \mathbf{x}_v, \mathbf{c}_q \|^{\frac{-2}{\beta-1}}}{\sum_{q=1}^{P} \| \mathbf{x}_v, \mathbf{c}_q \|^{\frac{-2}{\beta-1}}} \tag{5.7}$$

$$m_\gamma(L) = 1 - \sum_{q=1}^{P} m_\gamma(l_q),\ \gamma = 1, 2, \ldots, K \tag{5.8}$$

where q is the class number, u_{vq} is the fuzzy membership grade of $\mathbf{f}_v$ associated with class $\boldsymbol{l_q}$ and is used to measure the belief of $\mathbf{f}_v$ belonging to class $\boldsymbol{l_q}$, and ζ is a fixed factor used to normalize the total mass function.

Since there are *K* nearest neighbors $\{\mathbf{f}_\gamma, \gamma = 1, 2, \cdots K\}$ of $\mathbf{f}_v$, each of them can be treated as a piece of evidence supporting our belief that $\mathbf{f}_v$ belongs to class l_q. By using the *Dempster Rule of Combination* [28], we can fuse the mass functions of all k-nearest neighbours $\mathbf{f}_\gamma$ belonging to the same class $\boldsymbol{l_q}$ to form a combined mass function through orthogonal sum of the mass functions, represented as $m_{<q>}(l_q) = m_1(l_q) \oplus m_2(l_q) \oplus \cdots \oplus m_K(l_q)$. Based on the mass functions that are assigned to class q for all training data connections $\{\mathbf{f}_\gamma, \gamma = 1, 2, \cdots, K\}$ in the *K* nearest neighbors, the combined mass functions for the data sample assigned to class q and the whole frame L are given, respectively, by:

$$m_{<q>}(l_q) = 1 - \prod_{r=1}^{K} (1 - m_\gamma(l_q)) \tag{5.9}$$

$$m_{<q>}(L) = \prod_{r=1}^{K} (1 - m_\gamma(l_q)) \tag{5.10}$$

The difference between our work and the classifying algorithm in [24] is that, as shown in (5.9) and (5.10), all *K* neighbors rather than a subset of the *K* neighbors contribute to the belief that the test sample belongs to class l_q, making the classification more tolerable to training/test data variations.

A global mass function is further defined by considering all possible classes for the test data sample $\mathbf{f}_v$ in estimating the belief value of the sample belonging to class l_q. Hence, the global mass functions $m_v = m_v(l_1) \oplus \cdots m_v(l_q) \cdots \oplus m_v(l_P)$ of the test sample belonging to class l_q and the whole frame L are modified as:

$$m_v(l_q) = \frac{m_{<q>}(l_q) \prod_{r \neq q} m_{<r>}(L)}{H} \tag{5.11}$$

and

$$m_v(L) = \frac{\prod_{q=1}^{P} m_{<q>}(L)}{H} \tag{5.12}$$

where H is the normalizing factor, given by:

$$\begin{aligned} H &= \sum_{q=1}^{P} m_{<q>}(l_q) \prod_{r \neq q} m_{<r>}(L) + \prod_{q=1}^{P} m_{<q>}(L) \\ &= \sum_{q=1}^{P} \prod_{r \neq q} m_{<r>}(l_q) + (1-P) \prod_{q=1}^{P} m_{<q>}(L) \end{aligned} \tag{5.13}$$

The belief function *Bel* is widely used to measure the credibility of a hypothesis in classifying a test data sample. One can assign the mass function in (11) to $Bel_v(l_q)$ as the probability of the input data sample $\mathbf{x}_v$ belonging to class l_q. In this work, considering the inaccuracy and randomness in test/training data, we will apply the pignistic probability, which includes a measure of plausibility for more tolerance of test/training data inaccuracy in the classification. The pignistic probability *BetP* for an input sample $\mathbf{x}_v$ belonging to class l_q is defined as:

$$BetP_v(l_q) = m_v(l_q) + \frac{m_v(L)}{P}, q = 1,2,\cdots,P \tag{5.14}$$

6.2.2.1 Second-Stage Classification for Unknown Attacks by Neutral Zone

There may have a case where none of the belief values have a confident possibility of belonging to any of the existing class labels. In other words, some of the connections are too ambiguous to be assigned to any of the existing classes. In this situation, we may want to classify these connections into a new class label by using neutral zone [16].

After our FCM k-NN DST classifier process, the ideal classification for belief value to the correct corresponding class set is further defined. When the maximum pignistic probability of $m(l_{q\max})$ of the sample connections is more close to value 1, it means these connection samples are definitely belonging to the class l_q. Those values of $m(l_{q\max})$ that are more close to value 0 might not belong to any of the class labels l_q. Those values of $m(l_{q\max})$ that fall between 0 and $g(0 < g \leq 0.5)$ has more possibility of not being a member of any of the existing class but that of a new type of unknown attack class. The $m(l_{q\max})$ value of 0.5 is the crossover point of neutral zone data set. Any $m(l_{q\max})$ value greater than 0.5 implies that the connection definitely is the member of the class label l_q. As the $m(l_{q\max})$ value goes below 0.5, it is less likely that the connection is a member of any of the existing class label l_q.

In order to further evaluate the quality of DST belief assignment, the difference between the two maximum belief values $m(l_q)\ where\ (0<\mathrm{m}(\mathrm{l}_q) \leq 0.5)$ of each network traffic connection to the certain class can be considered. If the value of this difference is large a connection can be assigned clearly to one of the class. If the value of this difference is small a connection belongs to both classes to the same degree and its assignment is very

ambiguous. In this case, new network traffic connections with ambiguous class distribution need further classification to get a better result. Neutral zone concept is applied here to separate these connections as unknown attacks from the attacks result of output data from the FCM Weighted k-NN DST classifier.

For example, A={(a 0), (b 0.5), (c 0.7)} and B={(a 1), (b 0.5), (c 0.3)} are fuzzy sets in which a, b, and c have membership degrees in the set of A of 0, 0.5, and 0.7, and B of 1, 0.5, and 0.7 respectively. So it's absolutely true that a belongs to class B. But b and c are only partial members in the class A or B. Especially b with the same membership to class A and B which will cause the vagueness and insufficient of evidence of the classification. Note that the output classifier defined by our classifier is ambiguous if for two different output classes belief value for l_q has the same value or very close value.

To avoid this ambiguous classification in a case where the belief values are not equal but very close to each other, the uncertainty margin $\lambda(0 < \lambda < 1)$ can be explored here. This ambiguous belief values of new connections to all classes indicate that this connection does not belong to any of the existing known classes, and it belong to unknown attack classes. Let $m_v(l_{q\max})$ and $m_v^{'}(l_{q\max})(\forall q = 1, \cdots p)$ denote the two largest belief values of the sample traffic connection *v*. If the two largest memberships of some vectors are close to each other, we group these vectors to the neutral zone as the class of unknown attacks. Mathematically, instances for a single vector *v* can be characterized by the following condition in the flowing equation.

$$m_v(l_{q\max}) - m_v^{'}(l_{q\max}) \le \lambda, where 0 \le \lambda \le 0.5 \qquad (5.15)$$

We use λ as user-defined threshold to further determine the output data of FCM Weighted k-NN DST classifier belongs to the neutral zone subset (where unknown attacks class has been classified). It might not be desirable to mix all the output vectors from the result of our classifier by using different classifier just to decrease the detection rate or generate false alarms. The problem is how to obtain the threshold λ. In this paper $\lambda$is specified through a fixed percentage of the neutral zone traffic connections by our FCM K-NN DST classifier. For example the first classifier only passes with 35% of the data for further classification of unknown attacks. In order to avoid the problem of imbalanced data sets in KDD 99, Difference threshold is assigned to different classes (in our case, five different λ for five classes).

6.2.2.2 Second-Stage Classification for Unknown Attacks by Entropy Function

Unlike regular intrusion detection algorithms in which the incoming data are classified as either the normal data or one of the known attacks based on the maximum likelihood of all known classes, in this work we will introduce another second-stage intrusion detection mechanism to identify the input data as either normal data, one of known attacks or unknown attack to compare with the Neutral Zone method as the second-stage classifier. The first stage detection is the same with last section to identify if the input data are the normal data or an attack based on the pignistic probability of each class hypothesis. If the class type of the maximum pignistic probability is the normal data (l_{norm}) or the following equation holds,

$$l_{norm}(\mathbf{x}_v) = \arg\max_q BetP_v(l_q) \tag{5.16}$$

the input data connection $\mathbf{x}_v$ is considered to be a normal data connection. Since the classifier is fully and reliably trained with the labelled normal data, if (5.16) is true, the classification result becomes final and no further test is needed. However, if (5.16) is false, the input data are either one of the known attacks or a novel attack with unknown features and the following second-stage entropy-based test is needed to determine the attack type of the incoming data.

$$E_v = \sum_{q=1}^{P} BetP_v(l_q) \log_2 \frac{1}{BetP_v(l_q)} \leq \mu \tag{5.17}$$

where μ is predetermined threshold between (0, 1). If the hypothesis in (5.17) is true, i.e. the entropy of the generated pignistic probabilities is relatively small, the input data connection $\mathbf{x}_v$ is strongly correlated to one of known attacks. Therefore, $\mathbf{x}_v$ is considered to be one of the known attacks, and its class index q^* in the class set is given by:

$$q^*(\mathbf{x}_v) = \arg\max_q BetP_v(l_q) \tag{5.18}$$

However, if (5.17) is not true, the classification result is not credible and the input data are an unknown attack. The decision tree of the two-stage fuzzy classifier for known and unknown intrusion detection is shown in Figure 5.2.

5.3 Experimental Results

We still used DARPA KDD99 [29] Intrusion Detection Evaluation dataset as a part of the benchmark for evaluating the performance of the proposed IDS in detecting known intrusions. In addition, we have generated an unknown attack dataset from Software *Wireshark* [30] to test the performance of the new algorithm for detecting unknown attacks.

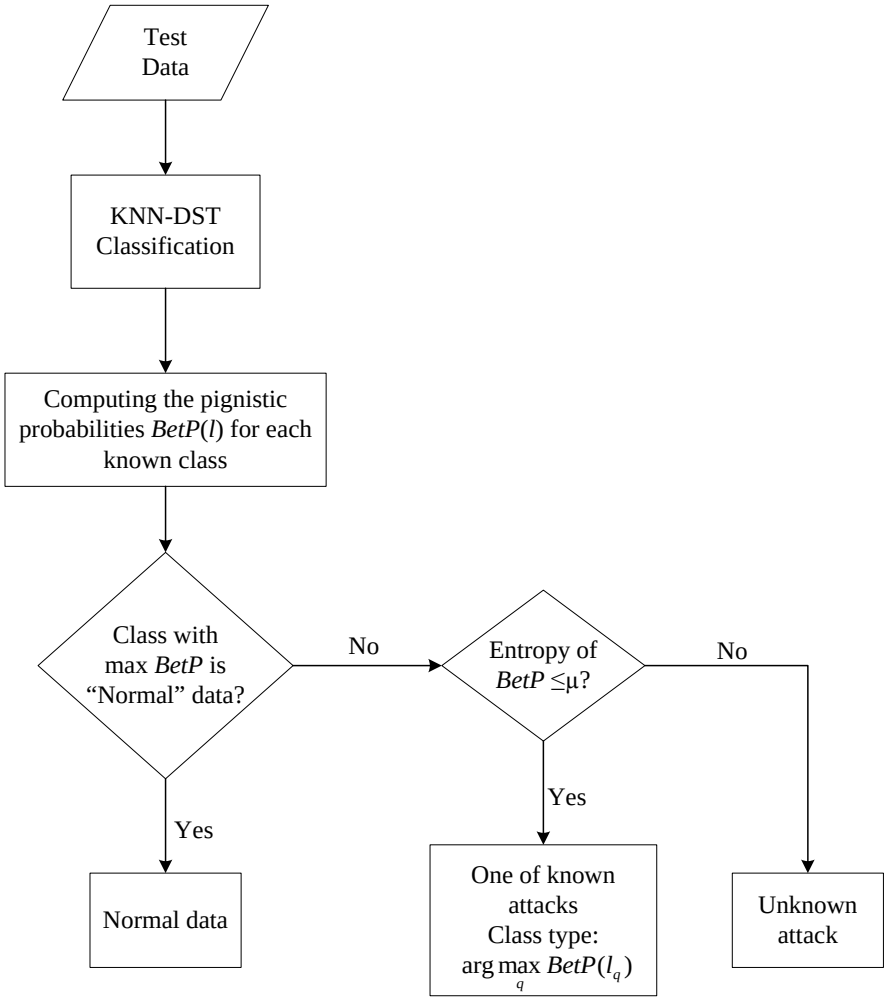

FIGURE 5.2 DECISION TREE OF TWO-STAGE CLASSIFIER

5.3.1 Dataset Selection

In this experimental work, the 10% KDD dataset were used as the training dataset to train the FCM algorithm and finalize the optimal feature space centers of the known classes. Subsequently, the two-stage fuzzy classifier is built based on the training results. Finally the corrected KDD 99 dataset and the unknown attack data generated from *Wireshark* are employed to evaluate the performance of our new classifier.

We add 5074 novel attack connections in this chapter to test our classifier. We use Mac OS X version 10.7.5 as the victim machine to hit malicious web sites running exploit kits such as Blackhole, which will probe our victim computer and attempt to infect it. Other methods we use are to visit malicious web sites either by offering of free or to click on spam e-mail messages. After being infected by these attacks, the Wireshark software installed on our victim machine is then used to manage and monitor malware activity.

5.3.2 Data Sets Pre-processing

Besides the normal data class, the KDD 99 datasets contain four types of known attacks, including the denial of service attacks (DoS), the user to root attacks (U2R), the remote to local attacks (R2L), and the probing attacks (Probe). However, even for the concise 10%KDD dataset, as its attack distribution statistics shown in Table 5.1, the data sizes are still overwhelmingly large for practical training and test applications. In addition, the data records in KDD 99 for different attacks are not equally represented and those in the same class are unbalanced for training dataset (10%KDD) and the test dataset (Corrected KDD). For instance, the percentages of U2R and R2L attacks in the training dataset are 0.105% and 2.279%, respectively, while those in the testing dataset are 0.733% and 5.204%. The discrepancies in data sizes and class distributions could make the classification results unreliable.

Furthermore, in the original KDD datasets, there are invalid and duplicated data records that need to be pre-processed and removed for algorithm training and testing. Therefore, the cleaned and rebalanced KDD 99 datasets with a reduced and manageable size have been created for our experimental work by randomly sampling the 10% KDD and

Corrected KDD datasets, in which the redundant and invalid data records are removed. The statistics of the size-reduced KDD datasets we generated for this work are listed in Table 5.2. In addition to the data records generated from the KDD datasets, we used additional 5074 data records containing unknown attacks in the testing phase to test the performance of our classifier. The types of the unknown attacks are summarized in Table 5.3 and they include some of the recently created Internet viruses including adware, spyware and their variants. The features of the unknown attacks may or may not be represented by those of the known attacks in KDD datasets. The unknown attack data records are used to test the proposed classifier and evaluate its performance in detecting unknown intrusions without any training.

The feature space of each connection in the KDD99 datasets and the unknown attack dataset we generated is composed of 41 feature components. For the KDD training data and unknown attack dataset, there is additional labelling information indicating which type of class the connection belongs to.

5.3.3 Performance Evaluation

To minimize the variations of our experimental results, we randomly divide both our training and test datasets into 10 subsets of equal sizes, and then we apply the new classifier to each pair of 10 training-testing subsets to evaluate its performance. We measure the performances of classifiers based on false positive rate (FPR), detection rate (DR) and overall error rate (OER). FPR, DR and OER for detecting one of the known intrusions or an unknown attack (Class l) from a batch of validation connections are defined as follows.

$$\mathrm{FPR}_l = \frac{\mathrm{FP}_l}{\mathrm{FP}_l + \mathrm{TN}_l} \tag{5.19}$$

where FP_l is the number of the connections that are incorrectly classified as Class *l* and TN_l is the number of the connections that are correctly classified as a class other than Class *l*.

$$\mathrm{DR}_l = \frac{\mathrm{TP}_l}{\mathrm{TP}_l + \mathrm{FN}_l} \tag{5.20}$$

where TP_l is the number of the connections that are correctly classified as Class *l* and FN_l is the number of the connections that are incorrectly classified as a class other than Class *l*.

$$\mathrm{OER}_l = \frac{\mathrm{FP}_l + \mathrm{FN}_l}{\mathrm{TP}_l + \mathrm{TN}_l + \mathrm{FP}_l + \mathrm{FN}_l} \tag{5.21}$$

To compare the performance of the new classifying algorithm with those of the existing classifiers, we also apply several popular classifiers including the basic k-NN, evidence-theoretic k-NN, naïve Bayes, and neural network classifiers [31, 32] to the same dataset used by the proposed classifier. The classification results are displayed with the receiver operating characteristics (ROC) plots, i.e. DR vs. FPR for all classifiers. Specifically, ROC plots are shown in Figures 5.3-5.12, respectively, for detecting *DoS*, *Probe*, *U2R*, *U2L* and unknown attacks (*UA*) by using existing classifiers as well as the our proposed new classifier.

Figure 5.3-5.7 show the new fuzzy DST classifier with neutral zone can almost compete other existing classifier in Probe, U2R, and R2L attacks, while it failed to achieve competitive detection rate with other classifiers in DoS attack. Figure 5.8-5.12 show that

the new fuzzy DST classifier with entropy function we are proposing almost outperforms all other existing classifiers by achieving higher DR and lower FDR for all known and unknown attacks. Since *DoS* and *Probe* attacks usually reveal a sequential pattern that is different from normal connections, they can be relatively easily be differentiated from normal data records.

However, *U2R* and *R2L* attacks do not possess a similar sequential pattern, and they are embedded in the data portions of the packets and normally only appear in a single connection. Therefore, the detection of *U2R* and *R2L* attacks from normal connections is more challenging than identifying *DoS* and *Probe* attacks; the detection rates of *U2R* and *R2L* intrusions with existing classifiers have been mostly unsatisfactory. However, using the new classifier, as shown our experiments, the detection rates of *U2R* and *R2L* attacks are significantly improved.

Table 5.4 lists the overall error rates (OER) in detecting different known and unknown attacks by using our new classifier with entropy function and other existing classifiers. OER, as defined in (20), includes the effects of both DR and FPR for a classifier, therefore, is a better indicator of classification performance. The results in Table 5.4 show that the OER of the new algorithm with entropy function is significantly lower than those of other existing classifiers in detecting the known and unknown intrusions.

In implementing the two-stage fuzzy KNN-DST classifier we choose the predetermined threshold μ to be 0.85 in the second-stage entropy-based classification. The second-stage detection is used to determine if an attack is unknown or one of known attacks, and it is only needed if the first-stage detection result is an attack. The experimental results

demonstrate that the new classifier with entropy function is effective in identifying unknown attacks as well as detecting typical known attacks from normal data traffic.

5.4 Conclusions

An innovative two-stage fuzzy k-NN DST classifier has been developed for effective detection of unknown intrusions and the variants of known intrusions. The new algorithm overcomes the rigid requirement of feature vector similarity between the training data and the test data in current IDS by introducing fuzziness, “soft” distance-based neighbouring concepts, and the DST-based evidence fusion method into the learning and classification schemes.

Furthermore, the two-stage entropy-based classification approach is employed to identify unknown attack in the incoming connections without any pre-training data or labeled information for the attack. The robustness and effectiveness of the new approach are demonstrated by the application results of the new classifier to the traditional KDD99 intrusion data and the newly simulated data containing both known and unknown attacks. The experimental results also show that the new classifier outperforms the existing classification algorithms in identifying known and unknown attacks from network traffic.

TABLE 5.1 DATA RECORD CLASS DISTRIBUTION IN KDD 99 DATASETS

Data Class	Training Set	Testing Set	Training %	Testing %
Normal	97,277	60,593	19.69%	19.48%
DoS	391,458	229,853	79.24%	73.90%
Probe	4,107	4,166	0.83%	1.34%
R2L	1,126	16,189	0.23%	5.21%
U2R	52	228	0.01%	0.07%
Total	494,020	311,029	100%	100%

TABLE 5.2 REDUCED TRAINING AND TESTING DATASETS (UA: UNKNOWN ATTACKS)

Data Class	Total	Normal	DoS	Probe	U2R	R2L	UA
Training Sets	145,585	87,831	54,572	2,131	52	999	0
Testing Sets	51,041	47,913	23,568	2,682	215	2913	5074

TABLE 5.3 UNKNOWN ATTACKS (UA) USED IN TESTING DATASETS FOR THIS WORK

UA type	Number of connections	Percentage
ZeroAccess botnet	1988	39.18%
Adware	875	17.24%
Spyware	869	17.13%
Backdoor	412	8.12%
Hijacker	275	5.42%
Trackware	181	3.57%
Downloader	187	3.69%
Trojan	287	5.66%
All UA types	5074	100%

TABLE 5.4 THE OVERALL DETECTION ERROR RATES OF OUR METHOD AND OTHER IDS

(ET k-NN: evidence-theoretic k-NN; NN: Neural Networks; NB: Naïve Bayes)

Class	Ours Method	ET k-NN	k-NN	NN	NB
DOS	5.22%	6.87%	11.18%	8.39%	6.57%
Probe	3.90%	7.01%	12.07%	7.80%	6.21%
U2R	8.12%	20.47%	35.06%	8.33%	13.13%
R2L	9.98%	19.31%	24.60%	10.29%	10.45%
UA	11.25%	40.68%	47.94%	44.88%	44.90%

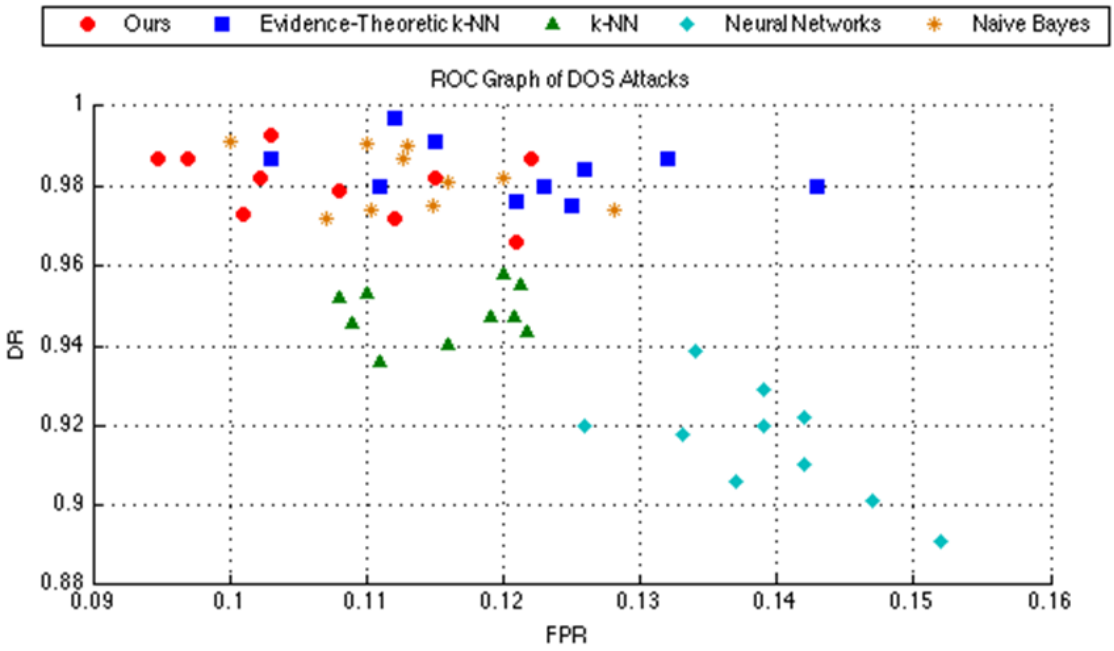

FIGURE 5.3 ROC GRAPH OF DOS ATTACKS USING THE NEUTRAL ZONE

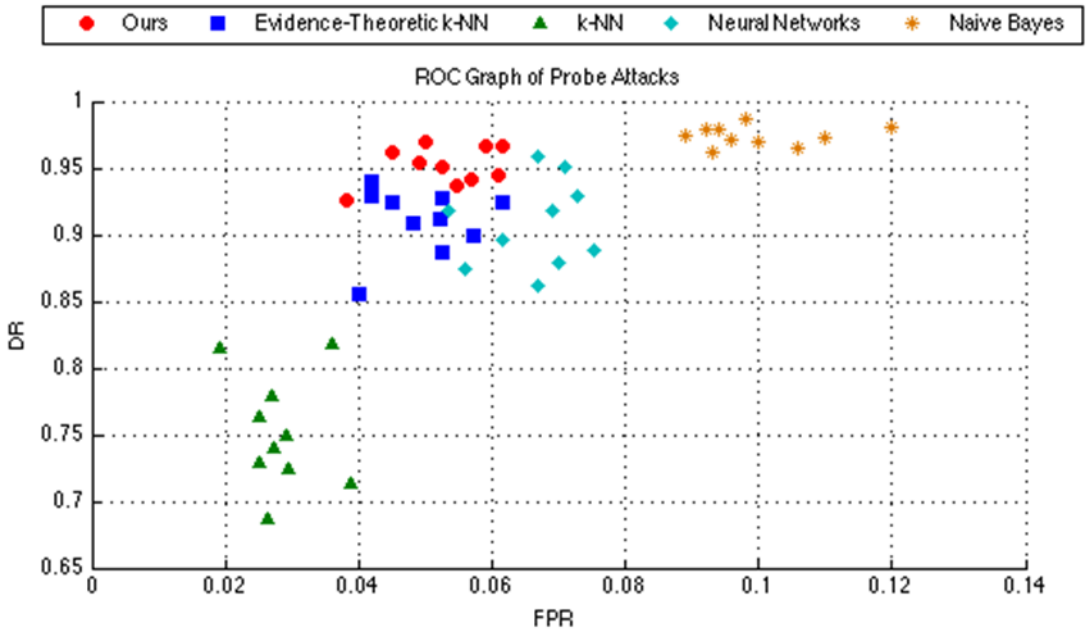

FIGURE 5.4 ROC GRAPH OF PROBE ATTACKS USING THE NEUTRAL ZONE

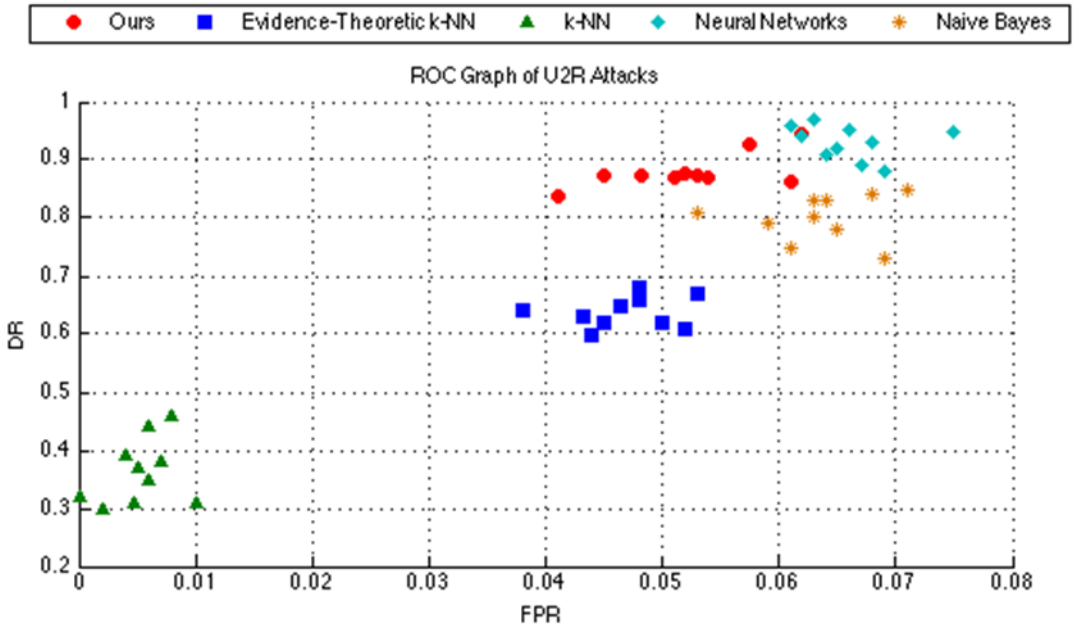

FIGURE 5.5 ROC GRAPH OF U2R ATTACKS USING THE NEUTRAL ZONE

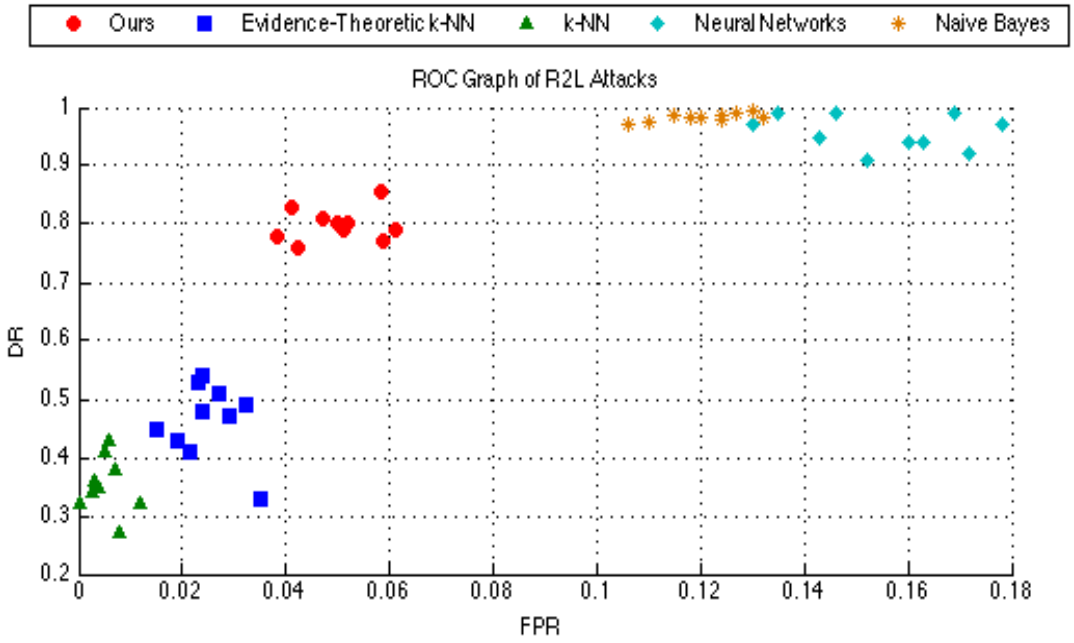

FIGURE 5.6 ROC GRAPH OF R2L ATTACKS USING THE NEUTRAL ZONE

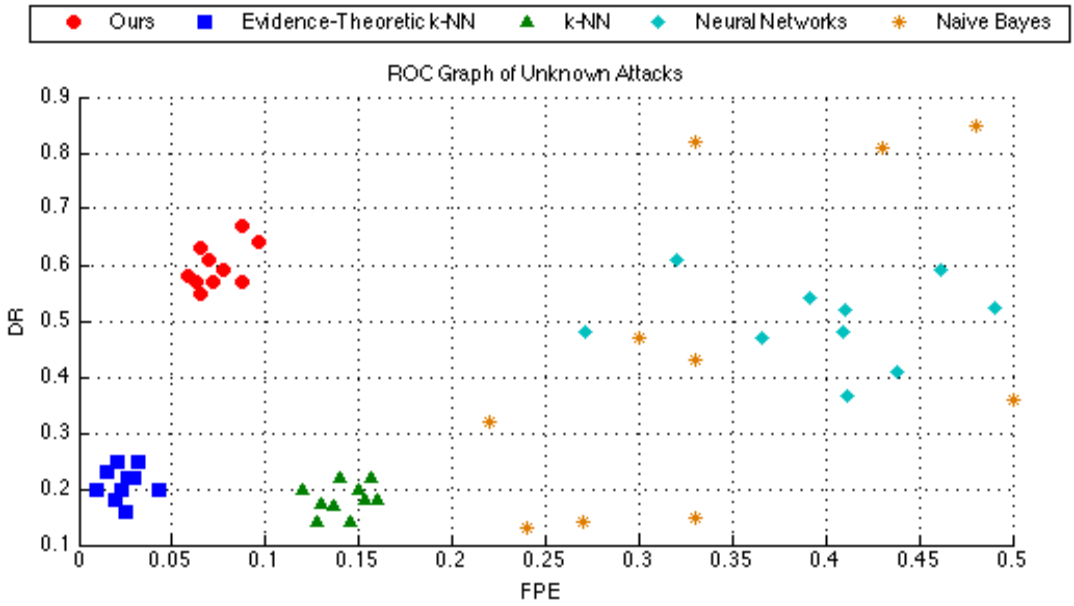

FIGURE 5.7 ROC GRAPH OF UNKNOWN ATTACKS USING THE NEUTRAL ZONE

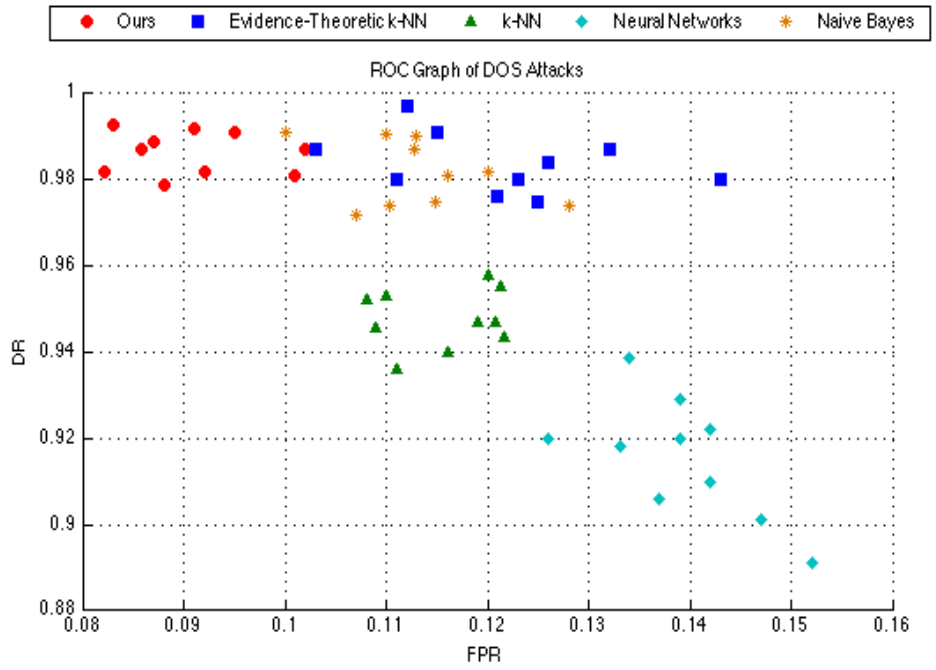

FIGURE 5.8 ROC PLOT OF DETECTING DOS ATTACKS USING ENTROPY

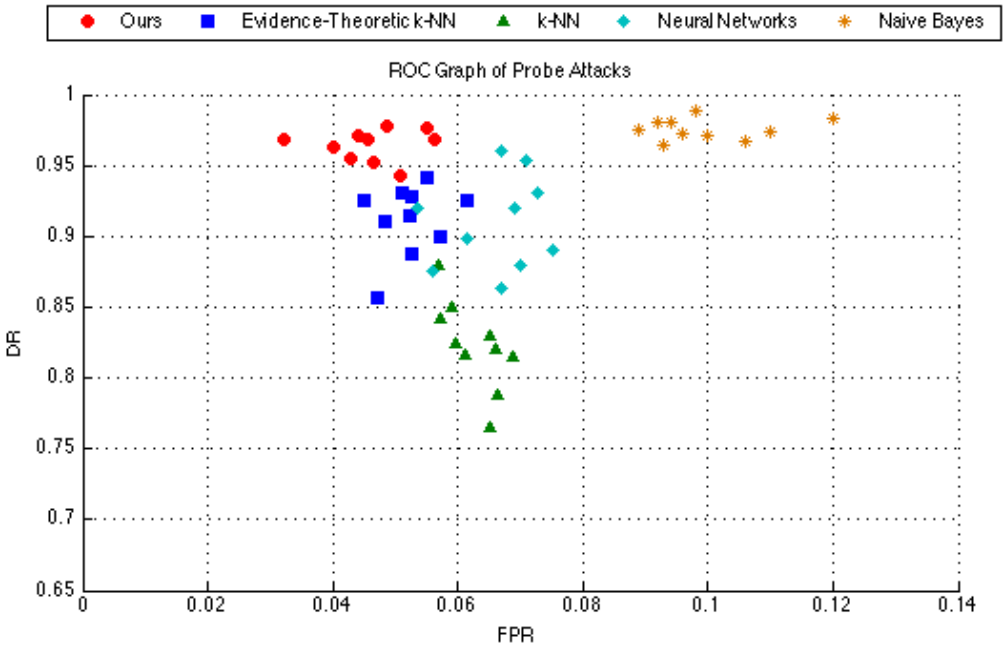

FIGURE 5.9 ROC PLOT OF DETECTING PROBE ATTACKS USING ENTROPY

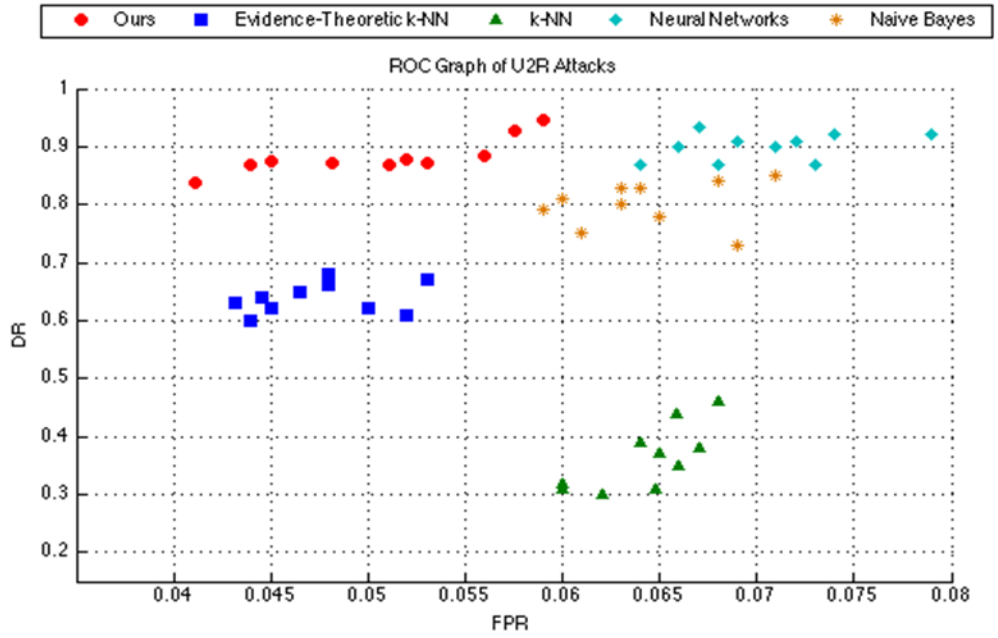

FIGURE 5.10 ROC PLOT OF DETECTING U2R ATTACKS USING THE ENTROPY

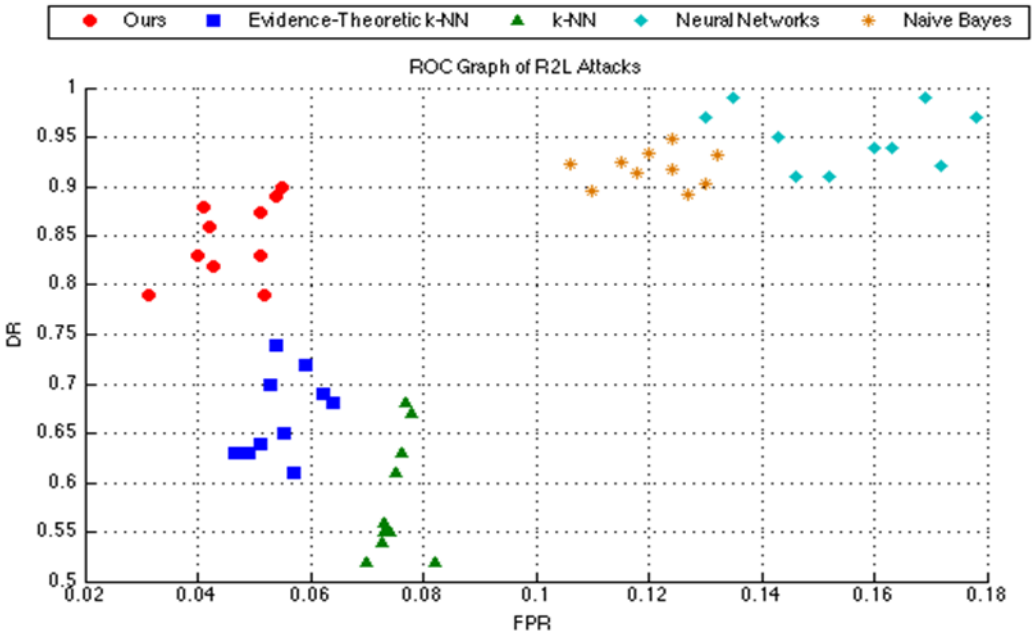

FIGURE 5.11 ROC PLOT OF DETECTING R2L ATTACKS USING THE ENTROPY

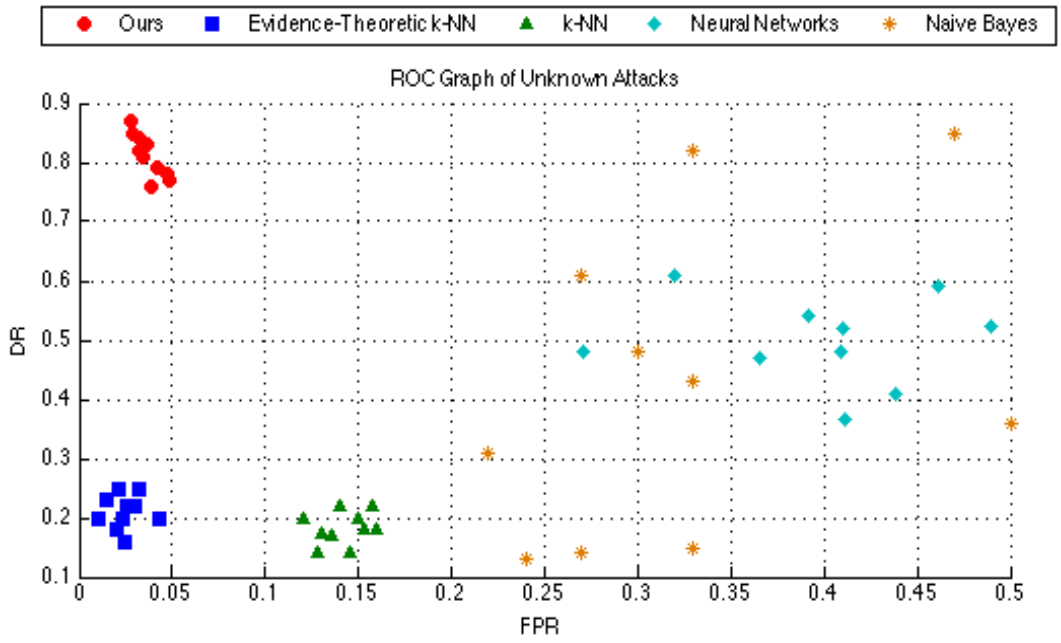

FIGURE 5.12 ROC PLOT OF DETECTING UNKNOWN ATTACKS USING THE ENTROPY

CHAPTER 6

THREE-LAYER HIERARCHY ENSEMBLE CLASSIFIER

In this Chapter, we improve the overall network intrusion detection rate by proposing and using an innovative three-layer hierarchy multi-classifier detection scheme called ensemble classifier. In addition, combinational methods are used to fuse the outputs from the classifiers are studied. Experiments show that ensemble-classifier using a diverse soft computing technique and different feature subset as a combination of multiple classifiers can obtain a much more precise inference result than a single classifier.

6.1 Introduction

Ensemble is to combine the outputs of a set of base classifiers together in a proper way when classifying input data. The fused result is expected to perform a better outcome than that of any individual base classifier within the ensemble. However, it is important to understand that individual base classifiers should be independent of each other. If the base classifiers provide similar outputs, then no significant improvement of the ensemble result can be obtained through the combination process. It is critical to notice the diversity among base classifiers in order to get effective and correct classification result. Hence, two major categories have been proposed in the ensemble classifier design. One uses different feature subset in every base classifier and the other uses different soft computing technique.

The former technique consists of a set of base feature selecting classifiers and each uses partial feature space. By choosing dissimilar feature subsets for various base feature

selecting classifiers, the diversity among these classifiers is expected to be maximized to achieve a better result. Example is the work of Giacinto and Roli [1]. In their research, they restricted the problem domain in the ftp service of the DARPA KDD99 data set and selected 30 out of the 41 available features from the data set features from the data set. They built three neural networks using 4 intrinsic features, 19 traffic features, and 7 content features, respectively. Also, they built one neural networks using all of the 30 selected features for the sake of comparison. All of the networks were three layers fully-connected multilayer networks, which each had 5 output neurons (for normal and four attack classes), a number of input neurons that equal to the number of features, and a hidden layer made up of 5 neurons for the networks using distinct features and 15 neurons for the network trained using 30 selected features. The results showed that the ensemble technique improved the overall detection performance compared with those of individual classifiers and the classifier using 30 features. However they only performed their experiments on ftp service instead of all of the services KDD99 data set provided. In the work of DeLooze [3], he created three 20X20 Self-Organizing Maps (SOM) using content, time, and connection features extracted from 41 features of KDD99 data set. The results of individual SOMs were then combined using both majority ensemble method and belief ensemble method. Here, the difficulty is how to configure a network with proper size. The configuration plays an important role in the detection performance and the granularity of the network nodes, which training a SOM with a large amount of neurons needs long computational time and a SOM with a small volume of neurons may loss some important information.

The work of Borji [4] is an example using different soft computing technique in every individual base classifier. He used KDD99 training data set in both training and test procedures as well as performed five-class (normal, DoS, Probe, U2R, and R2L) classification. He firstly used four base classifiers (neural networks, SVM, k -nearest neighbor (k -NN) and decision trees) to advance classification individually and then fused their inferences using three combination strategies: majority voting, average rule and belief function. He claimed his ensemble model overall got 99.68% detection rate (DR) and 0.87% false positive rate (FPR). However, he did not mention DR and FPR in each class. Also, we argue that if his experimental result still performed so well if KDD99 testing set was included in his experiment. The reason is that the testing set has extensive new types of attacks that are not correlated with attacks shown in the training set. Another example can be found in the work of Mukkamala et al. [5]. They designed two ensemble models: one consisted of three multilayer feedforward neural networks and the other was made up of neural networks, Support Vector Machine (SVM) and Multivariate Adaptive Regression Splines (MARS). By using the majority voting technique, the outcomes from individual base classifiers were then combined together. The experimental result showed that the ensemble approach produced a better result than that of each base classifier. In one of their experiments, they fused three base classifiers' outputs with 48%, 0%, and 16% accuracies together and get 56% ensemble accuracy. However, Hansen and Salamon [6] had proved that multi-classifiers will only work when it is possible to build individual classifiers which are more than 50% accurate. Furthermore, they used the same data set in both training and test procedures, which the experimental result cannot explain the detection ability of novel attacks.

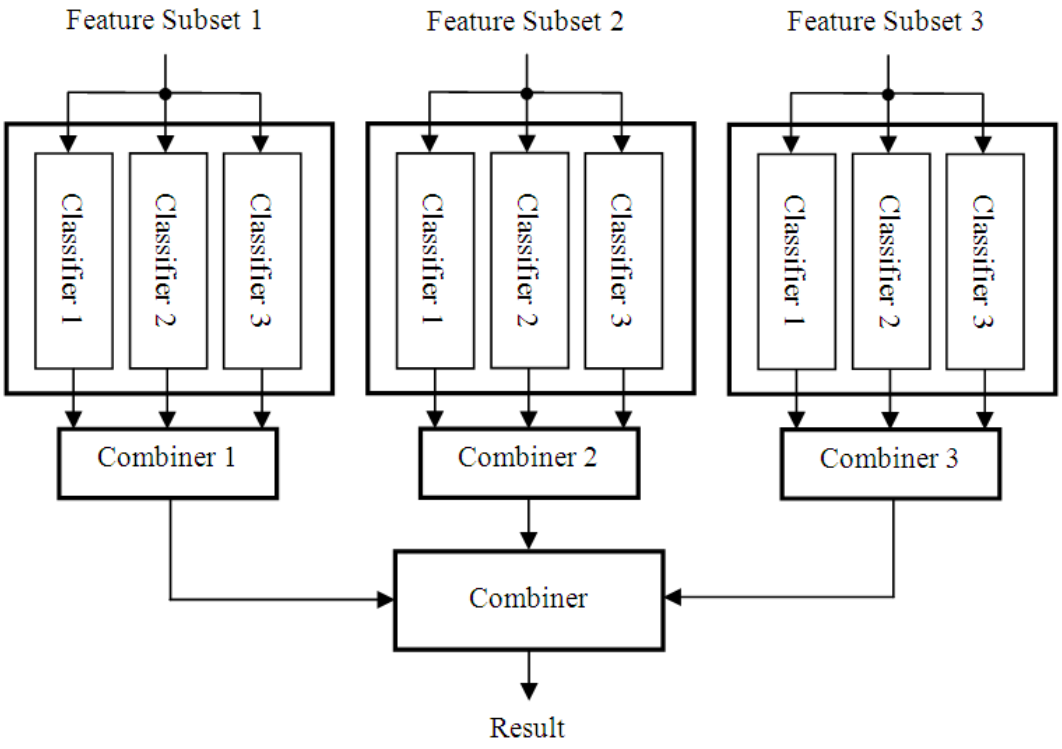

FIGURE 6.1 TOPOLOGIES OF PROPOSED INTRUSION DETECTION SYSTEM

6.2 Three-Layer Hierarchy Ensemble Classifier Approach

For a successful ensemble intrusion detection scheme, each classifier used in the system should be independent to others to achieve the best fusion result. Hence, we propose three-layer hierarchy multi-classifier intrusion detection architecture as illustrated in Figure 6.1.

In the first layer, three groups are constructed and each of them consists of a set of three base feature-selecting classifiers. In order to improve the diversity of three classifiers, different soft computing techniques as well as feature spaces are carefully selected and applied to the base feature selecting classifiers. In the second layer, the inferences derived from three base feature-selecting classifiers in a group are integrated and optimized.

Finally, the outputs from three groups are fused together to draw a final conclusion of the ensemble in the third layer.

There is often no clear boundary between normal and abnormal of a computer user's activity. Patterns of attacks are sometimes similar to those of normal activities. Therefore in the kernel of base feature selecting classifiers we select a variety of supervised learning techniques that can provide capability of dealing with vagueness: two-stage Fuzzy KNN DST classifier (proposed in chapter 5), naive bayes classifier, and backpropagation neural network classifier. All of them are capable of providing a dynamic decision boundary of network traffic instead of only assigning network traffic to a member of normal category or a member of abnormal category. During the entire course of work, the same data set KDD99 is used for training and testing those different soft computing models. For maximizing the diversity of the ensemble, three partial feature subsets, 9 basic features (1-9), 13 content features (10-22), and 19 traffic features (23-41), of the original KDD99 41 features are applied to three base feature selecting classifiers, respectively.

The major threats to sensor networks are the external attack and internal attack [2]. The malicious user concept arises originally from the requirements for open use systems. These unauthorized intruders are not uniformly cleared both for reasons of operational need and economy. The external attack is countered by using combinations of physical, procedural and communications security techniques. These techniques, some highly advanced, are the bulk of the present state-of-the-art in security issues. The internal attack is countered by using wormholes, sinkholes, select forwarding, and HELLO FLOOD attacks mostly through the routing process [3].

While we emphasize the threat from the malicious user, we are not unmindful of other security threats and risks. The problem of accidental leaking classified information, physical penetration of system sites, interference with or interrupt of communication, mishandling of classified material and the like are serious. To a large extent, these problems are common to any information system processing classified information and can be solved by well-understood techniques. But the intrusion prevention techniques (1st line of defense) like encryption/decryption, key management, and authentication alone cannot sufficiently protect the sensor networks from these attacks. In effect, the defence against these attacks surrounds the system and its user community with a barrier that must be breached before the system can be compromised. By adopting an effective intrusion detection system (IDS) as 2nd line of defence, the threat attacks can also be is eliminated before they get into the system [4]. Total cost of recovery from network epidemics like Nimda, Bagle, Code Red, SQL Slammer, and Mpvars attacks [6] are ginormous to the whole society. Especially in some circumstances, the accuracy of the classifiers is so crucial that a wrong prediction of attacks may result in extraordinarily high costs. Misclassifications [7] may come from the high similarities of the feature set between two objects, which make it difficult to distinguish them precisely.

6.3 Three Base Classifier For Three-Layer Ensemble Approach

We choose three base classifier for three-layer ensemble approach. Having finished the process of base feature selecting classifiers' derivations, all the decisions from multiple ones are combined into a fused result for each group. Finally, the predictions of three groups are then integrated to produce an ultimate conclusion of the ensemble.

6.3.1 Backpropagation Neural Network Classifier

A backpropagation neural network uses a feedforward structure to solve classification problems by its supervised learning algorithm. It consists of a collection of processing units that are highly interconnected. The network weights are updated by using gradient-based optimization algorithm during the training period. When the network converges to the local minima of error, the output layer of the network will show the result when data is fed into the input layer. Based on the data given for training, neural networks has the ability to learn how to process intrusion detection tasks. It acts as a computational model to process the network traffic information. It has the ability to generalize from learned data, and performs generalization of attacks and fault tolerance to imprecise and uncertain information. At the end of the training procedure, the future network traffic are then identified as whether malicious attacks or normal usage behavior.

6.3.2 Two-Stage Fuzzy KNN DST Classifier

The two-stage fuzzy KNN-DST classifier is proved to be capable to detect unknown intrusions and mutated versions of known intrusions. We choose to use the k-nearest neighbor (k-NN) algorithm for its robust performance and its tolerance for inaccuracy and random errors in the input data.

In this developed new algorithm, a belief value of a test connection associated with a known class is softly measured based on the distance between the input data and the centroid of the class, and the Dempster-Shafter Theory (DST) is incorporated into the framework to fuse multiple evidences generated from a weighted k-NN algorithm to form a pignistic probability of a test connection belonging to a known class. The centers of

known classes are softly defined and computed using a semi-supervised fuzzy c-means (FCM) algorithm from the training data.

Stage one classification determines if a connection is normal data or an intrusion. If it is an abnormal intrusion, stage-two classification is needed to determine if it is one of known attacks or unknown attack. We choose this classifier as our one of base classifiers to detect the unknown attacks especially.

6.3.3 Naive Bayes Classifier

The naive bayes classifier is based on conditional probabilistic to perform decision of a classification problem. It uses Bayes' Theorem with independence assumptions, which assumes a set of features are conditionally independent of one another given a class. When a set of classes are observed in the training data, the naive bayes classifier then assign an observed data to one of classes with highest probability. By applying naive bayes classifier to an intrusion detection task, a set of training network traffic data is given to find the prior probabilities for normal or a known class of attacks. As unseen network traffic arrives, the classifier then uses Bayes Theorem to decide which class the traffic should belong to.

6.4 Combination Methods For Three-Layer Ensemble Approach

In order to evaluate the result of different combination methods, we carried out four fusion techniques: the majority voting rule, the average rule, Dempster-Shafer technique, and Bayesian combination method. We discover our proposed approach to compare with classical ensemble models, such as boosting and stacking. Freund & Schapire demonstrate the use of boosting ensemble classifier to produce series of classifiers based

on their performance. The examples predicted incorrectly by previous classifier are chosen more often for a new classifier to learn. While our method will pass on the connections which are classified with low confidence for further classification and label classified connections with high confidence with correct class. Stacking is a technique by using meta-learner, which chooses more reliable classifier to get higher accuracy. Grading is another technique by using grading classification as meta-classes. Both arbitrating and grading use disagreements from references to select a new training set without generating new attributes.

Besides the notability of multiplicity among the base classifiers, the right choice of a combination method is also an important issue in creating a supreme performance. A variety of combination methods have been reported for combining the outputs of the base classifiers into an ensemble result. According to their characteristics, they can be classified as linear combination methods, non-linear methods, statistical-based methods, and computationally intelligent methods. Linear combination method is the simplest method to fuse base classifiers' outputs together. Summation and average are the popular ways for the combination. Non-linear method such as majority voting is used when the output of classifier is a ranked list of classes in accordance with the degrees of belief on classes the input pattern belongs to. Statistical-based methods are Dempster-Shafer techniques and Bayesian combination methods. The computationally intelligent method is based on computational intelligence techniques such as fuzzy logic, neural networks, and Naïve Bayes algorithms.

For comparing the performance of different combination operations in our intrusion detection task, we carry out four fusion techniques: the majority voting rule, the average

rule, Dempster-Shafer technique and Bayesian combination method to combine the outputs together. With equal posterior estimation distribution of classifiers' output, the majority voting rule assigns the input network traffic to the majority class among the outputs of classifiers. The average rule assigns the input network traffic to the maximum value of the posterior probability summation divided by the number of classifiers we implemented. As for the Dempster-Shafer and Bayesian combination methods, both assign the input network traffic to the class with highest belief value. The difference between them is that the Bayesian combination method involves the computation of the prior probability of each class but Dempster-Shafer technique does not, while it computes the probability that evidences support the attack or normal classes we consider.

6.5 Experimental Methodology

6.5.1 The Data Set

In our experiment, 10% KDD is taken as our training set and corrected KDD and 5074 novel attacks are taken as our testing set, respectively. KDD 99 data sets are made up of a large volume of network traffic connections describing TCP connections and each includes 41 features plus a label of either normal or a type of attack. The training set includes 494,020 connections that are distributed as 97,277 normal connections, 391,458 DoS attacks, 4,107 Probe attacks, 52 U2R attacks, and 1,126 R2L attacks. The testing set has two parts 311,029 KDD connections and 5027 novel attacks. The 311,029 KDD is made up of 60,593 normal connections, 229,853 DoS attacks, 4,166 Probe attacks, 228 U2R attacks, and 16,189 R2L attacks. The 5074 novel attack connections is added to test our classifier. We use Mac OS X version 10.7.5 as the victim machine to hit malicious

web sites running exploit kits such as Blackhole, which will probe our victim computer and attempt to infect it. Other methods we used are to visit malicious web sites either by offering of free or to click on spam e-mail messages. After being infected by these attacks, the Wireshark software installed on our victim machine is then used to manage and monitor malware activity.

6.5.2 Preprocessing

In the beginning of the experiment, we reduce the sizes of the original KDD 99 training and testing sets by removing the duplicated connections. The new training set has 145,585 connections that are distributed as 87,831 normal connections, 54,572 DoS attacks, 2,131 Probe attacks, 52 U2R attacks, and 999 R2L attacks. The new testing set has 51,041 connections that are distributed as 47,913 normal connections, 23,568 DoS attacks, 2,682 Probe attacks, 215 U2R attacks, and 2,913 R2L attacks.

For each connection, features represented by symbolic values are replaced by numeric values. For example, the values of icmp, tcp, and udp of feature protocol_type are replaced by values 1, 2, and 3, respectively. Values of each feature are normalized from 0 to 1 in order to offer equal importance among features. Class labels, normal, DoS, Probe, R2L, U2R, and novel attack are replaced by 1, 2, 3, 4, 5, and 6 respectively. In addition, a class label with values 1 and 2 is added to indicate normal traffic and attacks (DoS, Probe, R2L, and U2R), respectively.

6.5.3 Data Selection

Although the KDD99 data set includes 39 different types of attacks, the problem of uncertainty exists caused by limited information of network traffic data. In real world

modern computer systems and networks, hackers constantly develop new attack codes to exploit security vulnerabilities of organizations every day. It is impossible to cover all intrusive behavior space especially unknown attacks in the collected data set.

Accordingly, in order to simulate the problem of uncertainty, only a small amount of normal and attack connections are randomly selected from training and testing sets in each experiment. In the training set, all the 52 U2R attacks and 999 R2L attacks are included. For balancing the distribution of normal traffic and each attack group, 999 connections are randomly selected for normal class and each attack group (DoS, Probe, and U2R). In the testing set, all the 215 U2R attacks are included. Also, 215 connections are randomly selected for normal class and each attack group (DoS, Probe, and R2L).

6.6 Experimental Results

For detecting the attacks, training and testing are performed in each trial. In the training phase, three classifiers, two-stage fuzzy k-NN DST classifier, backpropagation neural network classifier, and naive bayes classifier, are constructed using the training data. The testing data are then fed into each trained classifier to identify normal behavior and intrusions in the testing phase. For two-stage fuzzy k-NN DST classifier, three nearest neighbors are selected for each testing connection. For the backpropagation neural network classifier, numbers of hidden neurons within each neural network is decided by the number of input features, which is equal to the square root of number of input features multiply by two.

We evaluate the performances of intrusion detection tasks by using standard measurements such as detection rate (DR), false positive rate (FPR), and classification

rate (CR). To minimize the inaccuracy and variation factor of experiment results, 10 trials are performed in every detection task and then the average of those trials is recorded.

Table 6.1 shows the averaged DR and FPR performances of three classifiers in each group of the first layer, which classifiers 1, 2, and 3 represent two-stage fuzzy k-NN DST classifier, backpropagation neural network classifier, and naive bayes classifier, respectively. The groups 1 stands for 9 basic features, and the group 2 stands for the 13 content features, and the group 3 stands for 19 traffic features are used, respectively.

The results indicate that the two-stage fuzzy k-NN DST classifier using content feature set has the best performance compared with those of other classifiers using partial feature set. It has very low FPR (1.33%) and DR (92.55%), which implies only few normal connections or malicious attacks are classified into normal behavior. By using group 1 of basic features set, two-stage fuzzy k-NN DST classifier again has the best overall performance, which its CR reaches 89.03% and its FPR is only 6.23%. By using group 3 of traffic features set, two-stage fuzzy k-NN DST classifier compete with other two base classifier with its CR of 90.07% and FRP of 4.33%.

For the backpropagation neural network classifier using basic feature set, it has both high FPR (93.21%) and DR (94.16%), which represents most of the connections are classified into attack group. For the Naïve Bayes classifier using content feature set, it has both high FPR (88.50%) and DR (13.44%), which represents most of the connections are classified into attack group. In general, the performances of two-stage fuzzy k-NN DST classifier using three diverse partial feature sets are equally well compared with those of the other two classifiers.

TABLE 6.1 THE PERFORMANCE OF THREE FEATURE SELECTING CLASSIFIERS

		Group 1			Group 2			Group 3		
		DR	*FPR*	*CR*	*DR*	*FPR*	*CR*	*DR*	*FPR*	*CR*
Layer 1	Classifier 1	96.59	6.23	89.03	92.55	1.33	94.57	94.21	4.33	90.70
	Classifier 2	94.16	93.21	76.69	85.98	13.72	86.04	83.49	10.28	84.73
	Classifier 3	63.53	3.35	70.16	88.50	13.44	88.11	65.47	1.07	72.16

TABLE 6.2 THE PERFORMANCE USING MAJORITY VOTING AND AVERAGE RULE

		Majority Voting			Average Rule		
		DR	*FPR*	*CR*	*DR*	*FPR*	*CR*
Layer 2	Combiner 1	85.70	16.56	85.25	89.02	16.74	87.87
	Combiner 2	88.74	13.58	88.28	52.79	7.16	60.80
	Combiner 3	80.35	5.44	83.19	80.00	4.74	83.05
Layer 3	Final Result	87.21	5.26	88.72	85.03	2.19	87.59

TABLE 6.3 THE PERFORMANCE USING DEMPSTER-SHAFER AND BAYESIANS

		Dempster-Shafer			Bayesian		
		DR	*FPR*	*CR*	*DR*	*FPR*	*CR*
Layer 2	Combiner 1	88.60	16.74	87.53	90.55	17.95	88.85
	Combiner 2	15.14	0.37	32.04	90.12	13.81	89.33
	Combiner 3	77.17	4.93	80.75	86.74	11.53	87.09
Layer 3	Final Result	83.49	1.91	86.41	93.35	9.63	92.75

Table 6.2&6.3 shows the performances of combiners on layers 2 and 3 using different combination methods. The results show that all of the four fusion techniques improve the overall performances in FPRs, DRs, and CRs compared with those of individual classifiers using partial feature sets shown in Table 6.1. For evaluating the performance

of the proposed ensemble model, the experiments of three classifiers using the entire 41 features are also done and the results are demonstrated in Table 6.4

The result indicates that all of the three classifiers using full feature set have equivalent CRs. All of their FPRs are below 11% and all the DRs do not reach to 85%. It also shows all of the four combination methods outperform the three classifiers using full feature set. Especially, the Bayesian combination method achieves the best outcome, which FPR, DR, and CR are 9.63%, 93.35% and 92.75%, respectively. Table 6.3 shows a comparison of three classifiers using full feature set and approaches using four different combination methods.

Consequently, we further analyze its detection accuracies of five attack groups with other classifiers and Table 6.5 shows the result. From the values we observe, the ensemble approach using the Bayesian combination method performs well in detecting DoS, Probe, U2R, and unknown attacks that each one has over 93.5% DR, R2L attacks with over 83.24% DR.

6.7 Summary

Ensemble-classifier technique has been applied to the intrusion detection task. We developed a three-layer hierarchy structure that includes three groups and each of them consists of three base feature selecting classifiers. In each base feature selecting classifier, we apply different machine learning algorithms and feature subsets to solve detection uncertainty problem and maximize the data diversity to achieve the best fusion result. During the experiments, we use a very small amount of network traffic data to simulate the limited information for the network embedded systems. Also, we compared the

performances of different combination methods in fusing the outputs derived from first and second layers in the proposed model. The experimental results have demonstrated that this hierarchy-structure method can achieve a better detection performance than that of a single classifier using either partial feature subsets or the full feature set. The result also shows that the Bayesian combination method achieves the best detection accuracy among the four diverse combination techniques. In addition the unknown attacks have been detected with good performance by using our proposed three-layer ensemble classifier.

TABLE 6.4 THE PERFORMANCE OF THREE FEATURE SELECTING CLASSIFIERS

		Group 1			Group 2			Group 3		
		DR	*FPR*	*CR*	*DR*	*FPR*	*CR*	*DR*	*FPR*	*CR*
Layer 1	Classifier 1	95.47	6.26	90.48	92.23	3.08	88.82	92.31	5.27	89.45
	Classifier 2	92.75	89.37	74.52	80.40	14.02	81.28	80.71	11.05	80.25
	Classifier 3	59.32	4.48	64.32	80.41	12.89	78.61	67.52	3.07	69.21

TABLE 6.5 COMPARISION RESULT

Method	**Normal**	**Probe**	**DoS**	**U2R**	**R2L**	**Unknown**
3 layer Ensemble (DR%)	99.85	99.36	**99.79**	**93.50**	**83.24**	**95.24**
3 layer Ensemble (FP%)	0.05	0.36	0.04	12.73	11.58	11.13
DT (DR%)	98.77	78.68	94.72	51.43	2.84	35.70
DT (FP%)	14.96	0.57	2.85	0.10	0.03	14.23
ID3 (DR%)	**99.93**	97.85	99.51	49.21	62.75	18.46
ID3 (FP%)	0.10	0.55	0.04	0.14	10.03	3.15
C4.5 (DR%)	98.3	**99.7**	76.3	21.1	30.2	42.90
C4.5 (FP%)	0.07	0.01	0.01	0.02	0.01	6.25

NB (DR%)	98.08	83.32	94.53	51.43	9.54	27.38
NB (FP%)	14.22	0.62	0.84	0.24	0.60	4.40
MLP (DR%)	99.6	75.5	99.71	14.3	32.7	33.15
MLP (FP%)	0.04	0.32	0.02	0.82	5.31	1.49
ESC (DR%)	98.2	84.1	99.5	14.1	31.5	27.32
ESC (FP%)	0.12	0.67	0.01	0.02	0.01	0.09
Boosting (DR%)	99.22	82.48	95.36	35.71	5.51	22.57
Boosting (FP%)	15.01	0.46	0.44	0.01	0.03	7.26
Bagging (DR%)	98.74	80.03	94.72	42.86	3.46	31.84
Bagging (FP%)	14.18	0.41	4.64	0.02	0.30	2.39
CFC (DR%)	99.03	97.46	88.24	60.00	21.47	16.22
CFC (FP%)	10.46	1.45	0.75	0.04	0.03	8.54

CHAPTER 7

CONCLUSION AND FUTURE WORKS

In this chapter, we summarize the research and then review the thesis contributions. At last, we discuss important future work.

7.1 Summary

This thesis describes an intrusion detection system for detecting computer intrusions from network traffic data. It consists of classification programs and rules, ensemble classifiers model, and two-stage classifiers.

We start with our research on studying the work in the field of intrusion detection systems. We describe the techniques used in designing intrusion detection systems as well as examine several representative approaches that have been implemented in intrusion detection systems. We then indicate that the data sources used by intrusion detection systems do have some problems, which are problem of irrelevant and redundant features, problem of uncertainty, and problem of ambiguity. These problems not only hinder the speed of detection but also decline the detection performance of intrusion detection systems.

We study the problem of uncertainty and ambiguity in audit network traffic data. The key idea is to imitate ambiguous of users' activities by fuzzy clustering technique, and to simulate uncertainty caused by limited information by incorporating only a small amount of network traffic data for analysis. Then, we identify future network traffic by fusing evidences found in clustering development by the use of Dempster-Shafer theory. Also,

we employ k-NN technique to speed up the detection process. We compare our result with those derived from three k-NN based unsupervised classification algorithms. The experimental results demonstrate that our approach has a superior performance to the other three algorithms. The results also demonstrate that our approach is capable of solving network traffic data which contain degrees of uncertain information.

We propose an ensemble intrusion detection model. We believe that the system is a preferable solution to achieve a higher detection performance with a combination of a set of base classifiers, which are built on the top of different feature subsets. In the ensemble intrusion detection model, In this paper, the ensemble-classifier technique is applied to the intrusion detection task. We develop a three-layer hierarchy structure that includes three groups of classifiers and each consists of three base feature selecting classifiers. In each base feature selecting classifier, we apply different machine learning algorithm and feature subset to solve uncertainty problem and maximize the diversity. During the experiments, we only include a very small amount of network traffic to simulate uncertainty caused by limited information. Also, we compare the performances of different combination methods in fusing the outputs derived from the first and second layers of proposed model. The experimental results demonstrate that this hierarchy structure obtain a better detection performance than that of a single classifier using either partial feature subset or full feature set. The result also shows that the Bayesian combination method achieves the best detection accuracy among those four diverse combination techniques. In the future, we will continue the research of further improving detection performance of both normal and malicious activities, especially in promoting the detection accuracy in R2L attacks

At last, An innovative two-stage fuzzy k-NN DST classifier has been developed for effective detection of unknown intrusions and the variants of known intrusions. The new algorithm overcomes the rigid requirement of feature vector similarity between the training data and the test data in current IDS by introducing fuzziness, “soft” distance-based neighbouring concepts, and the DST-based evidence fusion method into the learning and classification schemes. Furthermore, the two-stage entropy-based classification approach is employed to identify unknown attack in the incoming connections without any pre-training data or labelled information for the attack. The robustness and effectiveness of the new approach are demonstrated by the application results of the new classifier to the traditional KDD99 intrusion data and the newly simulated data containing both known and unknown attacks. The experimental results also show that the new classifier outperforms the existing classification algorithms in identifying known and unknown attacks from network traffic.

7.2 Thesis Contributions

We recap the thesis contributions as follows.

- We design a supervised machine learning algorithm that combines k-nearest neighbors technique, fuzzy clustering technique, and Dempster-Shafer theory. We apply this algorithm to intrusion detection task and successfully solve the uncertainty problems caused by deficient incomplete and ambiguous network traffic information.
- We propose an ensemble intrusion detection model that combines ensemble feature selection classifier and data mining classifier. It is a three-layer hierarchy structure, which each base classifier acts as an independent intrusion detector. By combining

these decisions from multiple base classifiers, this framework improves the detection performance.

- Unknown malicious attacks can be correctly classify by entropy function to improve the overall detection rate involving unknown abnormal behaviors in discovering intrusive behaviors.
- Semi-supervised learning is used to help predictive model testing to reduce the cost and improve the performance by using both labeled data (usually small amount) and unlabeled data (usually large amount) for training.

7.3 Future Work

Up to now, this dissertation has developed a network intrusion detection system based on ensemble of multiple base detectors. However, there are several interesting and important topics that need to be future explored.

- **False alarms:** In the design of anomaly intrusion detection systems, not only a high detection rate is necessary but also a low false alarms rate is required. However, it is not easy to control the false alarm rate because many unusual events sometimes are classified to hostile activities and most of these unusual events are actually normal behavior. In our research, we use data mining technique to extract decision rules from training set to reduce the number of false alarms. We believe that the false alarm rate can be further reduced in the future if a more dedicated rule set can be built. The solution could be achieved by applying a better data mining algorithm.
- **Respond to the intrusions:** In our work we focus on developing a detection method which can efficiently and effectively differentiate intrusive activities from large

volume of network events. We believe that the respond to the intrusions is also equally important. Once an intrusion is happened, it is necessary to properly present the alarm in order that system administrator can make correct and prompt decision. In a word, to find a method to integrate the intrusion detection system with the intrusion response system deserves further research.

- **Feature selection:** Feature selection plays an important role on both speed and accuracy of intrusion detection. It selects the most informative features that cover normal and intrusive activities by analyzing large quantity of network traffic data. In this dissertation we have developed a feature selection algorithm based on symmetrical uncertainty measure to remove the worthless information from the original high dimensional database. However, we think there are still some issues that can be explored in order that a better performance of our designed intrusion detection system can be obtained, e.g., relevant and redundant features analysis and discretization methods and correlation based methods implementation and comparison.
- **Multiple identification ability:** The *KDD99* data set includes four groups of attacks and each uses diverse skill to explore system's vulnerabilities. In our work we use binary classification technique to identify a network event as either normality or abnormality. The future direction could upgrade our system with multiple identification ability, i.e., the system can classify a network traffic data to normal activity or one of four attacks.

REFERENCES

[1] International Telecommunication Union. Ict statistics. http://www.itu.int/ITUD/ict/, Sept. 2012.

[2] H. Sedjelmaci, and M. Feham, “Novel hybrid intrusion detection system for clustered wireless sensor network”, International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.4, 2011.

[3] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford and N. Weaver, The Spread of the Sapphire/Slammer Worm, http://www.cs.berkeley.edu/~nweaver/sapphire/, 2011.

[4] S. Panigrahi, and S. Sural, "Detection of Database Intrusion Using a Two-Stage Fuzzy System", LNCS Volume 5735/2009, 107-120, DOI: 10.1007/978-3-642-04474-8_9, 2009

[5] N. Shrivastava, and V. Richariya, “Ant Colony Optimization with Classification Algorithms used for Intrusion Detection”, IJCEM International Journal of Computational Engineering & Management, Vol. 15 Issue 1, 2012

[6] S-Y. Wu and E. Yen. Data mining-based intrusion detectors. Expert Syst. Appl., 36, 5605–5612. ISSN 0957-4174. 2009.

[7] J. Cannady. Next generation intrusion detection: Autonomous reinforcement learning of network attacks. In Twenty Third National Information Security Conference, pages 1–12, Oct. 2010.

[8] S. Kumar, “Classification and detection of computer intrusions”, PhD Thesis, Department of Computer Sciences, Purdue University, USA, 1995.

[9] K. Q. Yan, S. C. Wang, S. S. Wang, and C. W. Liu, “Hybrid Intrusion Detection System for Enhancing the Security of a Cluster-based Wireless Sensor Network,” Proc. 3rd IEEE International Conference on Computer Science and Information Technology, Chengdu, China, pp.114-118, 2010.

[10] M. Mischiatti and F. Neri. Applying local search and genetic evolution in concept learning systems to detect intrusion in computer networks. In Proceedings of Workshop about Machine Learning and Data Mining. Seventh conference AI*IA "Intelligenza Artificiale", 2000.

[11] S. Forrest, A. Perelson, L. Allen, and R. Cherukury. Self-nonself discrimination in a computer. In Proceedings of the IEEE Symp. on research in security and privacy, 2010.

[12] K. Jones and R. S. Sielken, “Computer System Intrusion Detection: A Survey,” Tech Report, Computer University of Virginia, 2000.

[13] T. H. Hai, E. N. Huh and M. Jo, “A Lightweight Intrusion Detection Framework for Wireless Sensor Networks”, Wireless Communications and mobile computing, Vol.10, Issue.4, pp.559-572, 2010.

[14] E. Denning, “An Intrusion-Detection Model,” IEEE Transactions on Software Engineering, Volume 13, Number 2, pp. 222-232, 1987.

[15] S. E. Smaha, “Haystack: An Intrusion Detection System,” Fourth Aerospace Computer Security Applications Conference, pp. 37-44, Austin, Texas, 1988.

[16] J. D. Howard, *An Analysis of Security Incidents on the Internet 1989 – 1995*, Dissertation, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1997.

[17] M. Dekker, “Security of the Internet,” The Froehlich/Kent Encyclopedia of Telecommunications, Volume 15, pp. 231-255, New York, 1997.

[18] J. P. Walters, Z.Liang, W.Shi, and V.Chaudhary, “Wireless Sensor Network Security: A Survey”, Security in Distributed Grid and Pervasive Computing, Auerbach Publications, CRC Press, Vol.1, Issue.2, pp.1-50, 2006.

[19] A. Karygiannis, E. Antonakakis, A. Apostolopoulos, ”Detecting critical nodes for MANET intrusion detection,” 2nd Int. Workshop on Security, Privacy, and Trust in Pervasive and Ubiquitous Computing, pp. 7-15, 2009.

[20] S. Kumar, “Classification and Detection of Computer Intrusions,” PhD Thesis, Purdue University, 1995.

[21] J. Z. Lei and A. Ghorbani, “Network Intrusion Detection Using an Improved Competitive Learning Neural Network,” 2nd Annual Conference on Communication Networks and Services Research, p.p. 190-197, 2004.

[22] S. Zanero and S. M. Savaresi, “Unsupervised Learning Techniques for an Intrusion Detection System,” Proceedings of the 14th ACM Symposium on Applied Computing, 2004.

[23] URL: http://www.snort.orgS. Northcutt and J. Novak, *Network Intrusion Detection*, 2003.

[25] Dorothy E. Denning, D. L. Edwards, R. Jagannathan, T. F. Lunt, and P. G. Neumann. A Prototype IDES| A Real-Time Intrusion Detection Expert System. Technical report, Computer Science Laboratory, SRI International, 1987.

[26] Dorothy E. Denning and Peter G. Neumann. Requirements and Model for IDES { A Real-Time Intrusion Detection System. Technical report, Computer Science Laboratory, SRI International, August 1985.

[27] T. F. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. G. Neumann, H. S. Javitz, A. Valdes, and T. D. Garvey. A Real-Time Intrusion Detection Expert System (IDES) { Final Technical Report. Technical report, SRI Computer Science Laboratory, SRI International, Menlo Park, CA, February 1992.

[28] Teresa F. Lunt, R. Jagannathan, Rosanna Lee, Sherry Listgarten, D. L. Edwards, P. G. Neumann, H. S. Javitz, and A. Valdes. Development and Application of IDES: A Real-Time Intrusion-Detection Expert System. Technical report, SRI International, 1988.

[29] Sandeep Kumar. Classi_cation and Detection of Computer Intrusions. PhD thesis, Purdue University, West Lafayette, IN 47907, 1995. URL ftp://coast.cs.purdue.edu/pub/COAST/papers/sandeep-kumar/ kumar-intdet-phddiss.ps.Z

[30] Teresa F. Lunt, R. Jagannathan, Rosanna Lee, Alan Whitehurst, and Sherry Listgarten. Knowledge based Intrusion Detection. In Proceedings of the Annual AI Systems in Government Conference, Washington, DC, March 1989.

[31] Stephanie Forrest, Steven Hofmeyr, Anil Somayaji, and Thomas Longsta_. A sense of self for Unix processes. In Proceedings of the 1996 IEEE Symposium on Security and Privacy. IEEE Computer Press, 1996. URL ftp://ftp.cs.unm.edu/pub/forrest/ieee-sp-96-unix.ps.

[32] Teresa F. Lunt. Automated Audit Trail Analysis and Intrusion Detection: A Survey. In Proceedings of the 11th National Computer Security Conference, October 1988.

[33] Teresa F. Lunt. A Survey of Intrusion Detection Techniques. Computers & Security, 12(4):405{418, June 1993.

[34] Teresa F. Lunt, R. Jagannathan, Rosanna Lee, Sherry Listgarten, D. L. Edwards, P. G. Neumann, H. S. Javitz, and A. Valdes. Development and Application of IDES: A Real-Time Intrusion-Detection Expert System. Technical report, SRI International, 1988.

[35] L. Heberlein, G. Dias, K. Levitt, B. Mukherjee, J. Wood, and D. Wolber. A Network Security Monitor. In Proceedings of the IEEE Symposium on Research in Security and Privacy, pages 296{304, May 1990. URL http://seclab.cs.ucdavis.edu/papers/pdfs/th-gd-90.pdf.

[36] Judith Hochberg, Kathleen Jackson, Cathy Stallings, J. F. McClary, David DuBois, and Josephine Ford. NADIR: An automated system for detecting network intrusion and misuse. Computers and Security, 12(3): 235{248, May 1993.

[37] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, Tim Grance, L. Todd Heberlein, Che-Lin Ho, Karl N. Levitt, Biswanath Mukherjee, Douglass L. Mansur, Kenneth L. Pon, and Stephen E. Smaha. A System for Distributed Intrusion Detection. In Proceedings of COMPCON Spring '91, 36th IEEE Computer Society International

Conference, pages 170{176, San Francisco, CA, February 25 { March 1 1991. IEEE Computer Society, IEEE, IEEE Service Center, Piscataway, NJ.

[38] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur. DIDS (distributed intrusion detection system) - motivation, architecture, and an early prototype. In Proceedings of the 14th National Computer Security Conference, pages 167{176, Washington, DC, October 1991. URL http://seclab.cs.ucdavis.edu/papers/DIDS.ncsc91.pdf.

[39] Biswanath Mukherjee, Todd L. Heberlein, and Karl N. Levitt. Network intrusion detection. IEEE Network, 8(3):26{41, May/June 1994. URL http://seclab.cs.ucdavis.edu/papers/mhl94.pdf.

[40] Naji Habra, B. Le Charlier, A. Mounji, and I. Mathieu. ASAX: Software Architecture and Rule-based Language for Universal Audit Trail Analysis. In Proceedings of ESORICS 92, Toulouse, France, November 1992. URL ftp://coast.cs.purdue.edu/pub/doc/intrusion_detection/HabraCharlierEtAl92.ps.

[41] Phillip A. Porras and Peter G. Neumann. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In Proceedings of the 20th National Information Systems Security Conference, pages 353{365. National Institute of Standards and Technology, 1997. URL http://www.sdl.sri.com/emerald/ Emerald-NISS97.ps.gz.

[42] Abdelaziz Mounji, Baudouin Le Charlier, Denis Zampunieris, and Naji Habra. Distributed audit trail analysis. Technical Report RP-94-007, Institut d'Informatique, FUNDP, Rue Grandgagnage 21, Na- mur, Belgium, November 1994. URL ftp://coast.cs.purdue.edu/pub/doc/intrusion_detection/ MounjiCharlierEtAl94.ps.gz.

[43] Jai Sundar Balasubramaniyan, Jose Omar Garcia-Fernandez, Eugene Spa_ord, and Diego Zamboni. An architecture for intrusion detection using autonomous agents. Technical Report 98-05, COAST Laboratory, Purdue University, May 1998. URL ftp://coast.cs.purdue.edu/pub/COAST/papers/diego-zamboni/zamboni9805.ps.

[44] Steven R. Snapp, James Brentano, Gihan V. Dias, Terrance L. Goan, Tim Grance, L. Todd Heberlein, Che-Lin Ho, Karl N. Levitt, Biswanath Mukherjee, Douglass L. Mansur, Kenneth L. Pon, and Stephen E. Smaha. A System for Distributed Intrusion Detection. In Proceedings of COMPCON Spring '91, 36th IEEE Computer Society International Conference, pages 170{176, San Francisco, CA, February 25 { March 1 1991. IEEE Computer Society, IEEE, IEEE Service Center, Piscataway, NJ.

[45] Bruce Barnett and Dai N. Vu. Vulnerability assessment and intrusion detection with dynamic software agents. In Proceedings of the Software Technology Conference, April 1997.

[46] Stephanie Forrest, Steven Hofmeyr, Anil Somayaji, and Thomas Longsta_. A sense of self for Unix processes. In Proceedings of the 1996 IEEE Symposium on Security and Privacy. IEEE Computer Press, 1996. URL ftp://ftp.cs.unm.edu/pub/forrest/ieee-sp-96unix.ps.

[47] Stephanie Forrest. Personal communication, 1999. Department of Computer Sciences, University of New Mexico.

[48] Tim Bass. Multisensor data fusion for next generation distributed intrusion detection systems. In Proceedings of the IRIS National Symposium on Sensor and Data Fusion, May 1999. URL http://www.silkroad.com/papers/html/iris/.

[49] Tim Bass. Intrusion detection systems & multisensor data fusion: Creating cyberspace situational awareness. Communications of the ACM, April 2000. URL http://www.silkroad.com/papers/acm.fusion.ids.ps. Accepted for publication.

[50] Terran Lane and Carla E. Brodley. Temporal sequence learning and data reduction for anomaly detection. In Proceedings of the Fifth ACM Conference on Computer and Communications Security, pages 150{158. ACM, 1998. URL http://mow.ecn.purdue.edu/~terran/facts/research/pubs/acm_ccs98.ps.

[51] Kymie Tan. An application of neural networks to UNIX computer security. In Proceedings of the IEEE International Conference on Neural Networks, November 1995. URL ftp://coast.cs.purdue.edu/pub/ doc/intrusion_detection/neuralnetworks.ps.

[52] Steven Andrew Hofmeyr. An Immunological Model of Distributed Detection and Its Application to Computer Security. PhD thesis, University of New Mexico, May 1999. URL ftp://coast.cs.purdue.edu/pub/doc/ intrusion_detection/hofmeyer-distributed-detection.ps.gz.

[53] Jeremy Frank. Arti_cial intelligence and intrusion detection: Current and future directions. In Proceedings of the 17th National Computer Security Conference, Baltimore, MD, October 1994. URL http://seclab. cs.ucdavis.edu/papers/ncsc.94.ps.

[54] Stephanie Forrest, Steven Hofmeyr, Anil Somayaji, and Thomas Longsta_. A sense of self for Unix processes. In Proceedings of the 1996 IEEE Symposium on Security and Privacy. IEEE Computer Press, 1996. URL ftp://ftp.cs.unm.edu/pub/forrest/ieee-sp-96-unix.ps.

[55] Stephanie Forrest, Steven A. Hofmeyr, and Anil Somayaji. Computer Immunology. Communications of the ACM, 40(10):88{96, October 1997. ISSN 0001-0782. URL http://www.acm.org/pubs/citations/journals/cacm/1997-40-10/p88-forrest/.

[56] P. Garcia-Teodoro, J. Diaz-Verdejo, G.Macia-Fernandez, and E. Vazquez. Anomaly-based network intrusion detection: techniques, systems and chanllenges. Computer & Security, 28(1), 2009

[57] Heckerman D.A tutorial on learning with Bayesian networks. Tchnical Report MSRTR 95-06, Microsoft research, 1995.

[58] Wei Li. Using genetic algorithm for network intrusion detection. In proceedings of the United States Department of Energy Cyber Security Group 2004 Training Conference, pages 24-27, 2004.

[59] Ramadas M, Ostermann S, and Tjaden B. Detecting anomalous network traffic with self-organizing maps. Recent advances in intrusion detection (RAID), 2820:36-54, 2003.

[60] Qun Liu, Xingping Xian, Songtao Guo, Tao Wu, "Research on Cooperative Packet Forwarding and Punishment Mechanism in Wireless Sensor Networks" 2010 IEEE International Conference on Granular Computing

[61] Khairul Azmi Abu Bakar and James Irvine, "A Scheme for Detecting Selfish Nodes in MANETs using OMNET++", 2010 Sixth International Conference on Wireless and Mobile Communications 978-0-7695-4182-2/10 $26.00 © 2010 IEEE, DOI 10.1109/ICWMC.2010

[62] TEODOR-GRIGORE LUPU, "Main Types of Attacks in Wireless Sensor Networks" Recent Advances in Signals and Systems" 2011.

[63] DARPA. *SensIT - Sensor Information Technology*. http://www.sainc.com/sensit/goals.htm

[64] Smart-Its Consortium. *The Smart-Its Project*. http://www.smart-its.org/

[65] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci. *Wireless sensor networks: a survey*. Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 38, no. 4, pp. 393-422, March 2002.

[66] C. Chong and S.P. Kumar, *Sensor Networks: Evolution, Opportunities, and Challenges*. Proceedings of the IEEE, vol. 91, no. 8, pp. 1247-1256, 2003.

[67] C. E. Nishimura, D. M. Conlon. *IUSS dual use: Monitoring whales and earthquakes using SOSUS*. Mar. Technol. Soc. Journal, vol. 27, no. 4, pp. 13{21, 1994.

[68] *Proceedings of the Distributed Sensor Nets Workshop*. Pittsburgh, (USA), Carnegie Mellon University, 1978.

[69] R. Rashid, G. Robertson. *Accent: A communication oriented network operating system kernel*. ACM SIGOPS Operating Systems Review, vol. 15, no. 5, December 1981.

[70] R. T. Lacoss. *Distributed mixed sensor aircraft tracking*. Proceedings of the 6th American Control Conference, pp. 1827-1830, Minneapolis (USA), June 1987.

[71] V.R. Lesser, D.D. Corkill. *The distributed vehicle monitoring testbed: A tool for investigating distributed problem solving networks*. Readings from the AI magazine, pp. 69{85, ISBN: 0-929280-01-6, 1989.

[72] C.Y. Chong, K.C. Chang, S. Mori. *Distributed tracking in distributed sensor networks*. Proceedings of the American Control Conference, 1982.

[73] M. Panda and M.R. Patra. Network intrusion detection using naive bayes. IJCSNS International Journal of Computer Science and Network Security, 7, 258–263. 2007.

[74] I.H. Witten and E. Frank. Data Mining: Practical machine Learning Tools and Techniques. Morgan Kaufmann, 2nd edition, 2005.

[75] A. Estabrooks and N. Japkowicz. A Mixture-of-Experts Framework for Learning from Imbalanced Data Sets. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis*, pages 34–43, London, UK, Springer-Verlag. ISBN 3-540-42581-0, 2010.

[76] T.M. Khoshgoftaar, M. Golawala and J. van Hulse. An Empirical Study of Learning from Imbalanced Data Using Random Forest. In *ICTAI '07: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence - Vol.2 (ICTAI 2007)*, pages 310–317, Washington, DC, USA, IEEE Computer Society. ISBN 0-7695-3015-X, 2007.

[77] H. Guo and H.L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explorations Newsletter*, 6, 30–39. ISSN 1931-0145. 2006.

[78] Y. Xie, X. Li, E.W.T. Ngai and W. Ying. Customer Churn Prediction using Improved Balanced Random Forests. *Expert Systems with Applications*, 36, 5445–5449, 2009.

[79] S. Hido and H. Kashima. Roughly balanced bagging for imbalanced data. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 143–152, 2008.

[80] Y. Sun, M.S. Kamel, A.K.C. Wong and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, 40, 3358–3378, 2007.

[81] Y. Lu, "Knowledge Integration in a Multiple Classifier System," Application Intelligence, 6(2), pp. 75–86, 1996. http://public.itrs.net/Files/2003ITRS/Home2003.htm

[82] "Moore's law," (2007), Intel Corporation, [Online], Available: http://www.intel.com/technology/silicon/mooreslaw/

[83] K. Natori and N. Sano, ”Scaling limit of digital circuits due to thermal noise,” Journal of Applied Physics, vol. 83, pp. 5019-5024, 1998

[84] N. Sano, ”Increasing importance of electronic thermal noise in sub-0.1mm Si MOSFETs,” IEICE Transactions on Electronics, vol. 83, pp. 1203-1211, 2000

[85] L.B. Kish, ”End of Moore’s law: thermal (noise) death of integration in micro and nano electronics,” Physics Letter A, vol. 305, pp. 144-149, 2002.

[86] ”International technology roadmap for semiconductors,” (2003), ITRS 2003 Edition, [Online], available:

[87] N. Bashah, I. B. Shanmugam, and A. M. Ahmed, “Hybrid Intelligent Intrusion Detection System,” *Transactions on Engineering, Computing and Technology*, vol. 6, pp. 291-294, June 2005.

[88] M. Sabhnani and G. Serpen, “KDD Feature Set Compliant Heuristics Rules for R2L Attack Detection,” *International Conference in Computer Security and Management*, Las Vegas, Nevada, pp. 310-316, June 2003.

[89] W. Li, “Using Genetic Algorithm for Network Intrusion Detection,” *Proceedings of the United States Department of Energy Cyber Security Group Training Conference*, Kansas City, Kansas, CD-ROM Proceedings, May 2008.

[90] D. Song, M. I. Heywood, and A. N. Zincir-Heywood, “Training Genetic Programming on Half a Million Patterns: An Example from Anomaly Detection,” *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 225-239, 2009.

[91] K. M. Faraoun and A. Boukelif, “Neural Networks Learning Improvement Using the K-Means Clustering Algorithm to Detect Network Intrusions,” *International Journal of Computational Intelligence*, vol. 3 no. 2, pp. 161-168, 2008.

[92] L. Vokorokos, A. Balaz, and M. Chovanec, “Intrusion Detection System Using Self Organizing Map,” *Acta Electrotechnica et Informatica*, vol. 6, no. 1, 2009.

[93] 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, 2009.

[94] W. Lee and S. J. Stolfo, “A Framework for Constructing Features and Models for Intrusion Detection Systems,” *ACM Transactions on Information and System Security*, vol. 3, no. 4, pp. 227-261, 2010.

[95] S. Genaud, P. Gancarski, G. Latu, A. Blansch, C. Rattanapoka, and D. Vouriot. Exploitation of a parallel clustering algorithm on commodity hardware with P2P-MPI. *The Journal of Supercomputing*, 43(1):21–41, 2010.

[96] [Koscher 2010] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage. Experimental Security Analysis of a Modern Automobile, In Proc. of the IEEE Symposium on Security and Privacy, pp.447-462, 2010.

[97] W. Broad, J. Markoff J, and D. Sanger, Israel tests on worm called crucial in Iran nuclear delay. New York Times, January 15, p. A1. 2011

VITA

XUEYAN SHARON JING

1999-2003	B.A. Computer Information English
	Beijing University of Chemical Technology
	Beijing, China
2003-2004	Software Engineer
	Chinese Oil Trading Inc
	Shanghai, China
2005-2006	M.S Career & Vocational Technology Education
	California state University, San Bernardino
	California, CA
2008-Present	Doctoral Candidate, Electrical Engineering
	Florida International University
	Miami, Florida

PUBLICATIONS AND PRESENTATIONS

Xueyan Jing, Yingtao Bi, Hai. Deng, “An Innovative Two-Stage Fuzzy kNN-DST Classifier for Unknown Intrusion Detection”, The International Arab Journal of Information Technology. NO.4. July 2016

Xueyan Jing, Hai. Deng, “Neutral zone based fuzzy classification for intrusion detection”, IEEE Trans. Dependable Secure Comput., February 8-10, 2016 (submitted).

Xueyan Jing, Hai. Deng, "Intrusion detection on system-on-a-chip based sensor network", IEEE Trans. Wireless Commun., December 12-14, 2015 (submitted).

Te-Shun. Chou, Jeffrey. Fan, Sharon. Fan, Kia. Makki, "Ensemble of machine learning algorithms for intrusion detection", IEEE International Conference on Systems, Man, and Cybernetics (SMC'09), pp. 3976-3980, San Antonio, TX, October 11-14, 2009.

Sharon. Fan, Jeffrey. Fan, Kia. Makki, Nikki. Pissinou, "Communications via systems-on-chips clustering in large-scaled sensor networks", IEEE/IFIP 5th International

Conference on Embedded and Ubiquitous Computing (EUC'08), Shanghai, China, vol. 2, pp. 549-552, December 17-20, 2008

Te-Shun. Chou, Sharon. Fan, Wei. Zhao, Jeffrey Fan, Asad. Davari, "Intrusion aware system-on-a-chip design with uncertainty classification", IEEE 5th International Conference on Embedded Software and Systems (ICESS'08), pp. 527-531, Chengdu, China, July 29-31, 2008.

Chunchen. Liu, Rui-Xi. Chen. Jichang. Tan, Sharon. Fan, Jeffrey Fan, Kia. Makki, "Thermal aware clock synthesis considering stochastic variation and correlations", IEEE International Symposium on Circuits and Systems (ISCAS'08), pp. 1204-1207, Seattle, WA, May 18-21, 2008

Taoridi A. Ademoye, Asad. Davari, Charles C. Castello, Sharon. Fan, Jeffrey. Fan, "Path planning via CPLEX optimization", IEEE 40th Southeastern Symposium on System Theory (SSST'08), pp. 92-96, New Orleans, LA, March 17-18, 2008