



Intelligent Hybrid Approach for Classification Accuracy of Intrusion Detection System

Prepared By:

Mustafa Nihad Abbas

Supervisor

Prof.Dr.Mohammad Ahmad Alfayoumi

**This Thesis Submitted in Partial Fulfilment of the Requirements for
The Master Degree in Software Engineering**

Isra University

Amman, Jordan

2019/2020

AUTHORIZATION STATEMENT

I am Mustafa Nihad Abbas, authorize Isra University to provide hard copies or soft copies of my thesis to libraries, institution or individuals upon their request.

Name: Mustafa Nihad Abbas

Signature:



Date:

17/12/2019

اقرار تفويض

اني مصطفى نهاد عباس، افوض جامعة الإسراء للدراسات العليا بتزويد نسخ من رسالتي ورقياً و إلكترونياً
للمكتبات او المنظمات او الهيئات و المؤسسات المعنية بالأبحاث و الدراسات العليا عند طلبها.

الإسم: مصطفى نهاد عباس


التوقيع: 

التاريخ: ١٧ / ١٢ / ١٩٠٠

The undersigned have examined the thesis entitled (**Intelligent Hybrid Approach for Classification Accuracy of Intrusion Detection System**) presented by (**Mustafa Nihad Abbas**) a candidate for the degree of Master of information technology in software engineering and hereby certify that it is worthy of acceptance.

17/12/2019

Date



Prof. Dr. Mohammad Ahmad Alfayoumi

15.12.2019

Date



Dr. Mudhafar Al Jarrah

16.12.2019

Date



Dr. Venus W. Samawi

LIST OF CONTENTS

LIST OF CONTENTS	i
LIST OF TABLES	7
LIST OF FIGURES	8
ABSTRACT.....	10
CHAPTER ONE INTRODUCTION.....	1
1.1 OVERVIEW.....	1
1.2 PROBLEM STATEMENT	3
1.3 RESEARCH QUESTIONS.....	4
1.4 THE AIM OF THE STUDY	4
1.5 THE OBJECTIVES.....	4
1.6 THE SCOPE OF THE STUDY	5
1.7 RESEARCH PROCESSES	5
1.8 THESIS OUTLINE	7
CHAPTER TWO BACKGROUND AND LITERATURE REVIEWE	8
2.1 INTRODUCTION.....	8
2.2 INTRUSION DETECTION SYSTEM (IDS)	8
2.2.1 A brief history of IDS.....	10
2.2.2 Classification of IDS	11
2.2.3 Approaches to IDS	13
2.2.4 Challenges in IDS.....	16
2.3 Feature Selection	18
2.4 Feature Selection Evaluation Measures	20

2.5	FEATURE SELECTION PROBLEM IN IDS.....	23
	CHAPTER THREE DESIGN AND IMPLEMENTATION	26
3.1	INTRODUCTION.....	26
3.2	INTRUSION DETECTION SYSTEM (NSL-KDD).....	26
3.3	THE PROPOSED INTRUSION DETECTION SYSTEM.....	29
۳,۴	THE PROPOSED FEATURE SELECTION ALGORITHM.....	32
3.4.1	Firefly Algorithm	32
3.4.2	The Proposed FA.....	35
3.5	SUMMARY	39
	CHAPTER FOUR RESULTS AND DISCUSSION.....	40
4.1	Introduction	40
4.2	Experimental Settings	40
4.3	Results and Discussion.....	41
4.3.1	The Effect of Swarm Size and Number of Iterations	41
4.3.2	The Effect of <i>Swap</i> Variable	47
4.4	RESULTS COMPARTISON	49
	CHATER FIVE CONCLUSION.....	51
5.1	INTRODUCTION.....	51
	REFERENCES	55

LIST OF TABLES

Table 3-1	The features of the NSL-KDD data set.....	28
Table 3-2	Distribution of attack records per NSL-KDD attack category	29
Table 4-1	Parameter Settings	41
Table 4-2	The results of 15 run times for scenario 1	42
Table 4-3	The results of 15 run times for scenario 2	42
Table 4-4	The results of 15 run times for scenario 3	43
Table 4-5	The results of 15 run times for scenario 4	43
Table 4-6	The results of 15 run times for scenario 5.....	44
Table 4-7	The results of 15 run times for scenario 6	44
Table 4-8	The results of 15 run times for scenario 7	45
Table 4-9	The results of 15 run times for scenario 8	45
Table 4-10	The summarized results for the proposed algorithm	46
Table 4-11	Results of the proposed algorithm for Swap = 5	47
Table 4-12	Results of the proposed algorithm for Swap = 10	47
Table 4-13	Results of the proposed algorithm for Swap = 15	48
Table 4-14	Results of the proposed algorithm for Swap = 20	48
Table 4-15	The results of all the algorithms	50

LIST OF FIGURES

Fig 1-1	Research Process	6
Fig 2-1	The structure of IDS	9
Fig 2-2	Classification of IDS based on data collection and storage	11
Fig 2-3	Data analysis and process-based classification of IDS.....	12
Fig 2-4	Feature Selection Process	20
Fig 2-5	Types of feature selection evaluation measure.....	21
Fig 2-6	Filter-based feature selection.....	22
Fig 2-7	Wrapper-based feature selection	23
Fig 2-8	Process of Knowledge discovery.....	24
Fig 3-1	Block diagram of the proposed system	31
Fig 3-2	Flowchart of standard FA.....	34
Fig 3-3	Flowchart of GA-FA	36
Fig 3-4	The structure for each firefly.....	37
Fig 3-5	A graphical illustration for Crossover operator.....	38
Fig 4-1	The effect of swarm size and iteration number on the accuracy	46

LIST OF ABBREVIATION

IDS	Intrusion Detection System
DDOS	
R2L	
GA	Genetic Algorithm
GWO	Grey Wolf Optimizer
PSO	Particle Swarm Optimization
FFA	Firefly Algorithm
ACO	Ant Colony Optimization
NFL	No-Free Lunch Theorem
GD	Gradient Decent
KNN	K Nearest
SVM	Support Vector Machine
NSL	Network Socket Layer
BP	Backpropagation
PCA	Principle Components Analysis
RF	Random Forrest
SOM	Self-Organization Maps
SFLA	Shuffled Frog Leaping Algorithm
BBO	Biogeography-Based Optimization
PSO	Particle Swarm Optimization
ABC	Artificial Bee Colony
CS	Cuckoo Search
BA	Bat Algorithm

ABSTRACT

Intrusion detection system (I.D.S) is an essential component, which enhances the security of computer systems by actively detecting all forms of attack at the early stages. The main use of IDS is the monitoring of the network traffics and analyzing the behavior of the users in searching for any abnormal activity or attack signature for real-time intrusion detection. The main weakness in any IDS is their inability to offer adequate sensitivity and accuracy; coupled with their inability to process enormous data. To address these issues (such as the increasing traffic, huge behavior profiles, large signature databases, and the inability of differentiating normal behaviors from the suspicious ones), several algorithms have been developed. Hence, the main aim of this work is to choose the differentiating features for the development of an optimal machine learning algorithm which can offer high detection rates, fast training, and testing processes offline. The proposed machine learning model contains a feature selection algorithm (wrapper type) which is based on the integration of the Binary Firefly algorithm enhanced for feature selection by crossover operator taking from the genetic algorithm, called (GA-FA) with the Naïve Bayesian Classifier (NBC). The performance of the proposed model was tested on NSL_KDD data sets prepared by the MIT Lincoln Laboratory. The model testing was based on several experiments and different scenarios (the effect of swarm size, number of iterations, and the *Swap*). For evaluating the ability to select the minimum number of features with the higher value of classification accuracy, the algorithm worked perfectly and selected a comparable number of features. The model achieved the best average accuracy of 97.011%. In conclusion, the proposed feature selection algorithm has the ability to select the most relevant features which enhance the classification accuracy of the network intrusion detection system.

CHAPTER ONE

INTRODUCTION

1.1 OVERVIEW

The existing solutions are still incapable of providing full protection to internet and computer infrastructures against the ever-advancing network attacks despite the increasing awareness of the need for network security. It has become ever crucial to develop adaptive and effective security approaches for the internet and computer facilities. The first line security methods such as firewalls, user authentication, and data encryption have failed to sufficiently secure the entire network landscape due to the challenges of ever-evolving intrusion techniques and skills (Guan, Wang, and Zhang, 2009). It is therefore highly recommended to develop another line of security such as the intrusion detection systems (IDS). The IDS and anti-virus software have recently provided network security to the infrastructure of most organizations. The combination of these two defensive approaches offers an enhanced level of defense and efficiently protects the network against intrusions.

Intrusion detection involves the discovering and detection of network events or traffic on host machines which behaves abnormally or violating the network regulations. It helps in the analysis and monitoring of the daily activities within the computer systems to detect the presence of security threats. Meanwhile, the intrusion techniques are posing several security challenges to the security tools due to their sophisticated nature. Thus, an efficient and reliable IDS that will protect computer networks from all forms of attack is necessary. An accurate IDS that can discover network intrusions in real-time and fast enough in making decisions is required to solve these problems(Guan, Wang, and Zhang, 2009; Chen, Chen, and Lin, 2010).

The IDS can be generally grouped into the signature-based or misuse-based detection systems and anomaly-based detection systems. The signature-based systems detect current anomalies by searching for attack signatures that match any pre-defined attack (Vigna and Kemmerer, 1998). They are mainly used due to their simplicity and efficiency. Importantly, they generate a low rate of false-positive alerts. A major problem of these systems is the dependency of their efficiency and detection accuracy on the attack signature quality. Extracting such high-quality signatures requires the services of experts who have extensively studied malicious behaviors, and this might attract more cost and time. Additionally, an intrusion signature is required for the system to detect the respective intrusion. These systems do not detect previously unknown attacks due to the lack of attack signatures, thereby, making the networks protected by the signature-based systems to be vulnerable to unknown attacks. These weaknesses can additionally cause more critical problems in real practice because of the development of new sophisticated intrusion techniques which defeats the current security tools. Thus, there is a need to continuously update the signature database of these systems to detect new attacks.

Anomaly-based detection systems have attracted more research attention. These systems assume that the behavior of the intruders differs from normal network behavior (Hassanzadeh and Sadeghian, 2008; Chandola, Banerjee and Kumar, 2009). When compared to the signature-based systems, the anomaly-based systems are capable of detecting previously unknown attacks and other forms of known attacks since they statistically analyze the pattern of behavioral deviations of the monitored traffic flows from the behavior of normal traffic. They create normal flow models by studying normal network traffic behaviors. After that, they consider any deviation from the normal traffic flows as suspicious behavior. The major advantages of these systems are their ability to recognize known and unknown forms of attacks without the need for a continuous or steady update of the attack signature. However, their major disadvantages include the generation of a large number of false alerts in the presence of new normal network traffics as well as a poor detection efficiency when attacks mimic normal network behaviors. Similarly, they are not good at handling a large volume of data. Additionally, labeled data for the training of the detection models are usually not available, thereby, constituting a major issue (Davis and Clark, 2011; Altwaijry and Algarny, 2013).

Intrusion detection is a classification task, consisting of developing a predictive model with the capability of identifying attack instances. On the one hand, there are several attributes or features which may falsely correlate; the classification of the anomaly-based IDS is a complicated task due to the possibility of having many redundant features. As a result, the redundant features can be eliminated using feature selection methods without compromising the performance of the model (Amiri *et al.*, 2011). The IDS is usually run on a daily basis in the real world, and the instances in the IDS dataset are much and require much time to be classified or clustered. A dataset with over a million instances can take several days, large memory, and computation resources to be classified. Thus, feature selection is an important aspect of IDS since they usually include several instances and features (Lin *et al.*, 2012).

The determination of the relationship between features and class labels in an IDS dataset for the design of a feature selection algorithm is the main aim of this study. The features were selected from an IDS dataset using the developed feature selection algorithm. Furthermore, the performance of the selected features was analyzed and compared to other feature selection algorithms. Finally, the proposed algorithm was tested on other datasets to verify its effectiveness. In addition to the reduction of dimensionality and computation time, feature selection also eliminates irrelevant and redundant features from a dataset. An IDS dataset is usually got from network connections, and thus, usually, contains several useless and counterproductive classification features. With feature selection, these useless features are eliminated from the dataset before classification (Hansman and Hunt, 2005; Lin *et al.*, 2012).

1.2 PROBLEM STATEMENT

From the previous studies in this area, it is evident that the retraining of the reference models and the enhancement of the classifier's recognition capability determines the efficiency of an IDS model. For an IDS to have a better detection capability, feature selection is paramount. Feature selection involves the selection of feature subsets that represent the whole dataset. There are two aspects of the need for feature selection: the first is the filtering and removal of noise, and the second is the elimination

of redundant and irrelevant features which can cause a significant reduction in the accuracy of the model and time wastage during detection. In this study, a hybrid algorithm is proposed for selecting the most relevant subset of features, which enhances the classification performance of IDS. The proposed algorithm is Firefly Algorithm (FA) integrated with a Genetic Algorithm (GA).

1.3 RESEARCH QUESTIONS

There are two research questions for this study as follows:

- * How does the Genetic Algorithm (GA) enhance the global search ability of Firefly Algorithm (FA) for feature selection problems?
- * How to enhance the classification performance of Intrusion Detection System (IDS) using the proposed hybrid algorithm?

1.4 THE AIM OF THE STUDY

The development of a hybrid algorithm of the Firefly Algorithm with Genetic Algorithm for feature reduction and enhance of classification performance of Network Intrusion Detection System is the major aim of this study.

1.5 THE OBJECTIVES

There are three main objectives of this study:

- 1) To investigate the problems of network intrusion detection system (IDS) and study the trends of feature selection in IDS.
- 2) To design and develop a hybrid wrapper feature selection algorithm based on the Chaotic Firefly algorithm and Genetic algorithm for feature selection with Naïve Bayesian Classifier in an IDS.
- 3) To analyze and validate the developed algorithm using the NSL-KDD dataset.

1.6 THE SCOPE OF THE STUDY

The use of the Chaotic Firefly (CFA) and GA to select features that will represent the traffic classes that will be determined by the Naïve Bayesian Classifier for attack classification and improving the detection accuracy is the main domain of the study. The classification of attacks is based on two established dominant categories, which are ‘normal’ and ‘attack’, as widely used in other studies in the field of IDS. The study employed the NSL-KDD dataset which is widely used by other researchers.

1.7 RESEARCH PROCESSES

This methodology is comprised of five mandatory phases, comprising of a review of the literature, Firefly-Genetic Algorithm (GA-FA) design, (GA-FA) implementation, experiments and evaluation, and finally documentation. The flow chart of the methodology is represented in Figure 0-1.

In this research, a review of the existing works of literature is the first phase of the methodology. This review focuses more on presenting a general background on the Intrusion Detection System, and the application of machine learning models and feature selection algorithms in enhancing the performance of these systems. Despite the number of studies already conducted in this field, there are still gaps that need to be filled, suggesting the need for more efficient algorithms. The literature review also focused on the Firefly algorithm, the main source of inspiration and its own mathematical model.

The second phase of the study involved the designing of the FGA, the proposed new approach for hybridizing FA with GA and applying it as a feature selection method. The process flow was designed to portray the working of the proposed tool. Every aspect of the tools, starting from the input from the user until the system output, was designed prior to the implementation. Visual Studio.net 2019 IDE was used to implement the proposed algorithm,

while C#.net programming language was used to develop it. Next, after the implementation is the fourth phase where several experiments were carried out to verify the efficiency of the proposed feature selection method.

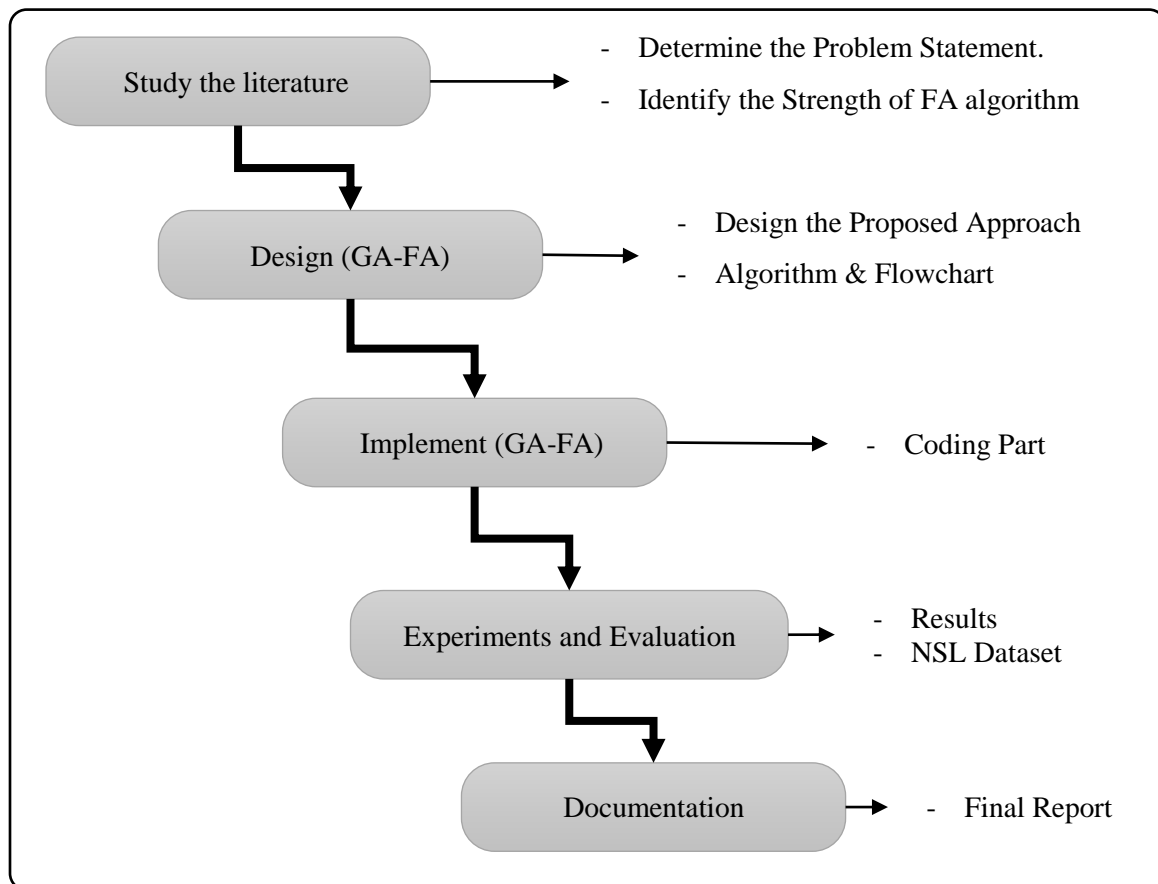


Figure 0-1 Research Process

The fifth phase of the research is the documentation phase which also works alongside phases 1-4 as well. All the processes starting from the literature review to the experimental results were documented at each phase. The final documentation involves the compilation of all the documents generated from each phase.

1.8 THESIS OUTLINE

Chapter 2 Theoretical Background: The background and introduction of the IDS and feature selection are presented in this chapter. The chapter also provided a comprehensive review of the recent feature selection methods.

Chapter 3 System Design and Implementation: the proposed feature selection algorithm is explained. The pseudocode with the flowchart is also given.

Chapter 4 Results and Discussion: The hybrid approach algorithm is examined and evaluated in this chapter.

Chapter 5 Conclusion and Future Work: summarizes the major contributions of the study and suggested the areas for improvements.

CHAPTER TWO

BACKGROUND AND LITERATURE REVIEW

2.1 INTRODUCTION

The major problem of the internet today in automatically detecting network attacks is the dynamic nature of the attacks and the ever-growing number of the target. IDS usually detect abnormal behaviors through packet monitoring. Machine learning technologies are often used to identify abnormal traffic behaviors, and there are two common machine-learning methods for attack detection, which are the classification-based and clustering-based methods. Some of these methods are not useful or lose their effectiveness when the data volume is large. Moreover, there are several features in the traffic data, and most of these features are either irrelevant or redundant. Therefore, feature selection algorithms are usually used to eliminate irrelevant and redundant features. In this chapter, the IDS, feature selection, and machine learning methods will be discussed. The background information is provided to guide the reader through the works presented in the subsequent chapters of this thesis.

2.2 INTRUSION DETECTION SYSTEM (IDS)

The IDS is used for the monitoring of malicious activities on networks; they also discover illegal actions and attack indications which have evaded the system security policy. In addition to helping the network administrator understand the attacks in a network system in real-time, a good IDS also provides the basis for constituting the policy of a network security system. Studies on the IDS started in the 1980s when Denning proposed the first IDS model in 1985[(Liao *et al.*, 2012)]. Since then, several studies

have been done on the construction of an effective IDS model. Because IDS is required for the protection, detection, and responding to system attacks, there is a need to consider several problems such as the identification of intrusion, data collection, reporting, and response when constructing IDS models. Figure 2-1 shows the structure of an IDS and its four basic components (Chandola, Banerjee and Kumar, 2012; Liao *et al.*, 2012).

1. Monitoring object: This can be a network or a host, and it is monitored.
2. Data collection and storage: here, all the network-related data is collected and converted to a proper format for storage.
3. Data analysis and management: In the IDS, this is a core part where the suspicious actions are searched and signals are generated in the presence of an attack. Having generated the signals, the IDS handles the attack or signals the network administrator to process the attack.
4. Signal: This may be regarded as the output of an IDS. It is an automatic response or an alarm to the network administrator.

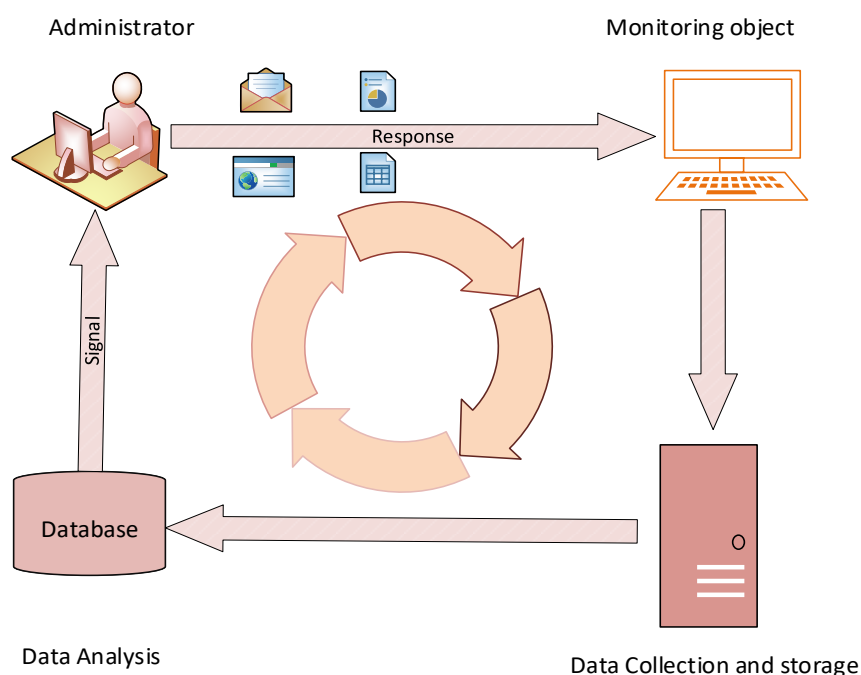


Figure 2-1 The structure of IDS

2.2.1 Brief History Of IDS

The concept of intrusion detection was first suggested by Anderson J P in 1980 who defined an intrusion as an attempt or threat that caused the unavailability of the system or unauthorized information access” (Chandola, Banerjee and Kumar, 2012). The idea of intrusion detection was proposed for the auditing of the record of the operating system, but researchers have concentrated less on this idea and rather focused more on the encryption and access denial to the data from authenticated hosts. Denning D E in 1985 suggested the IDS model known as the Intrusion Detection Expert System (IDES) which is made up of a host, an object, the audit record, the profile characteristics, an anomaly record, and the activity rules. This model relies less on the system platform, system weakness, application environment, and type of attack, and also offers a general IDS approach/framework. In 1988, Teresa L extended the model by creating a real-time IDS which can detect attacks as data when received. The Teresa model is employed to detect intrusions behavior for a single host. Heberlein (1990) suggested a network-based IDS when proposing the Network Security Monitor (NSM) that detects malicious activities by observing the network data in the local area network instead of checking the host’s audit record(Denning, 2012; Liao *et al.*, 2012).

Studies on the IDS have gradually increased since 1990. Some American research institutes have combined the host-based IDS with the NIDS to design a distributed IDS. Crosbie and Spafford improved the maintainability, scalability, efficiency and fault tolerance of IDS using autonomous agents. Sandeep Kumar studied the immune principle-IDS while Anderson introduced the information retrieval technology into IDS. Lane used the machine learning method to detect the anomaly behavior of users while Lee used the data mining technique on IDS(Vigna and Kemmerer, 1998; Chandola, Banerjee and Kumar, 2012; Ahmed, Naser Mahmood, and Hu, 2016).

2.2.2 Classification of IDS

The IDS can be divided into several groups based on the four parts presented in Figure 2-1. Based on different methods of data collection and storage, the IDS can be grouped into host-based IDS (HIDS), network intrusion detection system (NIDS), and application-based IDS. As shown in Figure 2-2, the NIDS regards raw network packets as a data source; the sensors collect the packets from a secure network to determine the status of the network. The response module alerts the administrator an attack is detected by the sensors. The HIDS was first developed in the 1980s but has not assumed the complexity of today's current models. It monitors packets based on a host and match packet behaviors with their signatures. The host-based IDS sensors obtain the audit data history from the operating system of the host. The application-based IDS operates on individual hosts as well, and their sensors gain log files from users' software and applications(Tsai *et al.*, 2009).

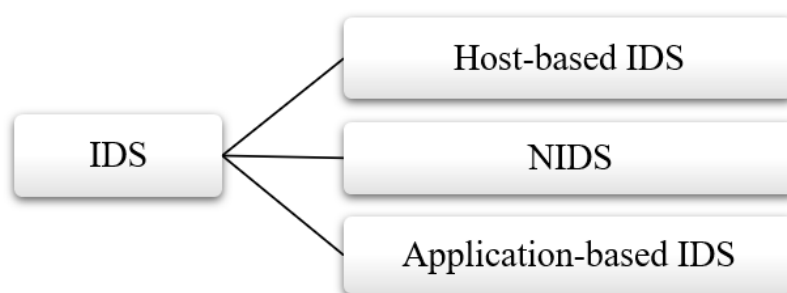


Figure 2-2 Classification of IDS based on data collection and storage

The IDS can be divided into misuse detection and anomaly detection based on different data analysis and processing units as shown in Figure 2-3. The misuse detection system is used for the analysis and detection of intrusion. They generally regard intrusion behaviors as a character or a pattern and establishes the characteristics of an intrusion mode based on already known intrusions patterns. They monitor the behavior of the system and match them with the established database. From the matching results, the system can determine an active intrusion. A supervised machine-learning approach can help in the composition of the signatures. The intrusion detection systems are highly effective in the detection of attacks they have been

programmed to detect but may fail to detect new attacks since they may not recognize them since they did not match their signature lists. The misuse-based IDS can be grouped into stateless and stateful misuse detection systems. The stateless misuse detection systems only employ the existing signature, but the stateful systems use not only the existing signatures but can utilize the previous signatures as well. Contrarily, the anomaly-based IDS can create a normal operation framework for the users in which an operation that does not match the normal behavior will be considered a threat. The principle of anomaly is based on taking every behavior as a possible attack and thus, can easily detect unknown attacks. The anomaly-based IDS can further be classified into self-learning and rule-based systems. Their difference is that the rule-based IDS has fully pre-specified normal rules but the self-learning IDS need to undergo a training process before being able to detect normal network behaviors(Tsai *et al.*, 2009; Aljawarneh, Aldwairi and Yassein, 2017; Kabir, Onik and Samad, 2017).

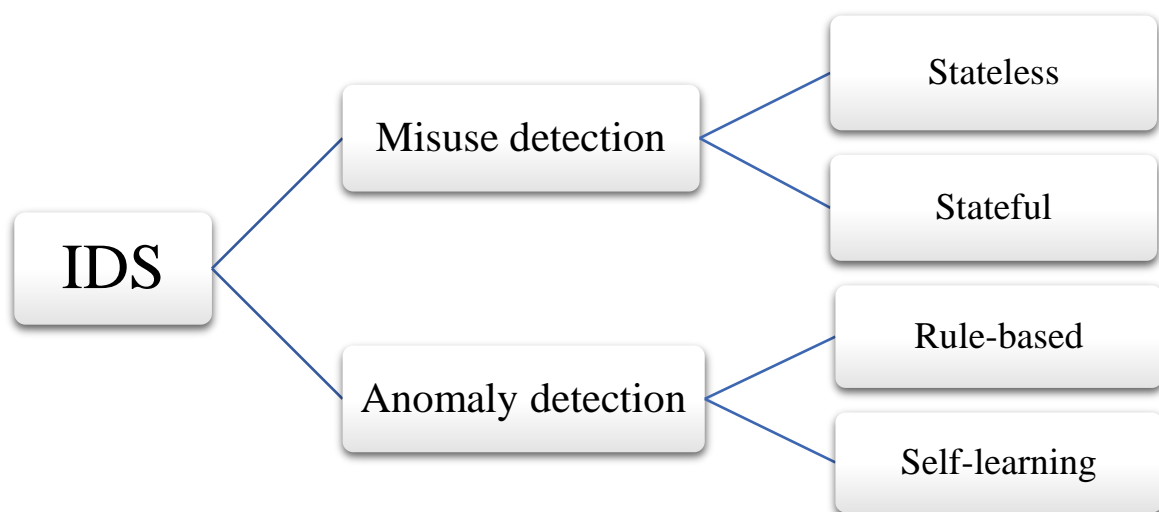


Figure 2-3 Data analysis and process-based classification of IDS

Based on the response mechanism, the IDS can be classified into the reactive response and passive response IDS, but based on the frequency of usage, they can be classified into offline and online IDS (Konstantinos and Lazaros, 2017).

2.2.3 Approaches to IDS

the IDS can be grouped in several ways. The commonest way of classification based on the data processing and analysis unit. As earlier stated, the misuse-based detection systems use the abnormal behavior rule to detect attacks, where actions that are consistent with the set rules are prevented. These rules are also referred to as signatures and are constructed using malicious patterns. The misuse detection system has a strong detection capability, though its major disadvantages are the need to always update the pattern database and the difficulty of detecting unknown attacks. The core of the misuse detection lies in the way to express attack behaviors and how to ensure the completeness of the intrusion actions. With the emergence of new network attacks and vulnerabilities, it may be difficult to reflect the database pattern of all possible attacks. This is the cause of the increase in the rate of false-negative alerts. There are three methods of misuse detection currently in use, which are as follows (Wu and Banzhaf, 2010; Davis and Clark, 2011; Chandola, Banerjee and Kumar, 2012):

1. Simple pattern matching

This is the commonest misuse detection method which has a high attack detection efficiency, ease of implementation, and strong performance in real-time, but can be used only in simple attack modes, and has a high rate of false-positive alert. Snort, the commonest tool for network intrusion detection employs the simple pattern matching which uses rules to describe known intrusion patterns. It can be modified, and also has a good readability.

2. Expert systems

This is one of the first systems for “misuse detection” which has been used in numerous classical intrusion detection frameworks. With the expert systems, the user provides the behavioral pattern known attacks to the system in a special format. The expert system then constructs the rules from the information and match the corresponding events and rules to determine the presence of an intrusion. To the users, the expert system is an autonomous "black box" and users need not interfere with their decision-making or internal reasoning processes. The main drawback of the existing expert system is the complexity of maintaining the rule, and the need to take into consideration the rule relationships when changing them. Another problem with the current expert systems is low efficiency while handling a large dataset(Tsai *et al.*, 2009; Liao *et al.*, 2013).

3. State transition diagram:

State transition analysis represents and detects known intrusions using state transition diagrams where an intrusion pattern is presented as a number of state transfer processes, while the process proceeds from an initial to a final state. The state transition analysis has the benefits of: first, they do not directly depend on the detailed data, rather identifies the important intrusion features that are supposed to be detected. Secondly, intrusions can be established before the completion of invasion in order to facilitate a timely response to the invasion. Thirdly, slow and cooperative attacks can be detected. Their drawbacks include: first, many intrusion patterns cannot be explained by the state transition diagram because it can only describe intrusion behaviors which might cause obvious systemic changes. Secondly, the intrusion behavior is described simply as a state sequence and some of the complex behaviors such as concurrency and conditions may not be described. Thirdly, the state transition diagram needs to determine the status of the system behavior before checking and matching it with the rules.

The anomaly-based detection systems judge an intrusion by establishing a clear difference between the monitored system and normal system behavior. The advantage of the anomaly detection systems is that having no need for much knowledge of the system defects. They are also strongly adaptable and can detect both new and unknown intrusion patterns. Their major drawback is the way to represent the normal user or system behavior. There are three methods of the current anomaly-based IDS(Mitchell and Chen, 2014):

4. Statistical method

The statistical anomaly-based detection IDS involves the use of a specific system or users' normal behavior statistical model for learning. It identifies abnormal behaviors as a deviation from the normal patterns based on statistical findings. The statistical method is mainly based on the statistical model and object selection, as well as statistical model training. Some of the possible statistical objects are as follows.

- A. User login and activity : This includes the frequency of user login, duration of the activity, number of password errors, etc.
- B. Execution of command and program : This includes the frequency of command execution, the use of the program running, etc.
- C. The file operation : This includes the files read, written, created, deletion frequency, as well as the operational failure frequencies.

The selection of the appropriate statistical model for specific intrusion detection in this method is easy and this can be regarded as an advantage of this system. Its weakness is the difficulty in the determination of the threshold value. The detection accuracy is influenced by the size of the values. Moreover, the description of most of the systems using a simple statistical

model or user behavior is difficult, while many sophisticated calculations are required in the statistical model.

1. Anomaly-detection based on the immune principle

The biological system for immunity against pathogens or the non-detection of the organization in itself is quite precise, and the "self/non-self" recognition is a basic and important function. Certain similarities between the computer protection system and the biological immune system have been established by researchers in the United State of America. This method uses the audit data through a fixed length of systems to describe the normal behavioral profile and the sequential distance as shown in the differences between the measured behavioral patterns. It has been shown experimentally that the immune principle-based anomaly detection system can detect several attacks that exploit program vulnerability. However, a major drawback of this system is its ability to detect only attacks that exploit program vulnerability(Bolzoni and Etalle, 2008).

2.2.4 Challenges in IDS

The intrusion detection systems were developed for distributed, high detection speed, intelligent, high accuracy and highly secure systems. Studies on the IDS mainly focus on the following(Davis and Clark, 2011; Chandola, Banerjee and Kumar, 2012; Liao *et al.*, 2012):

1. Distributed intrusion detection

The distributed intrusion detection systems are used mainly in the heterogeneous system and large networks where distributed structures are used for collaborative processing and information analysis. They are also used as a single IDS architecture compared to the greater detection ability.

2. Intelligent intrusion detection

The intelligent intrusion detection methods are the currently used methods, including machine learning, data mining and neural network methods. Several intelligent techniques have been investigated for the application of intrusion detection. The main purpose of this study is to reduce the rate of false alarms and to improve the self-learning capability and real-time response of the system. From the results of the present study, there are several advantages of the intelligent technology-based intrusion detection methods.

3. Protocol analysis-based intrusion detection

The intrusion detection rate based on the protocol analysis is small and this method can be employed for the detection of high degree network protocol regularity even in high load networks, even when packet loss generation is difficult.

4. Combined with the operating system

The rate of new attack detection by IDS can be enhanced when combined with an operating system.

5. Application layers intrusion detection

The semantic variables of many attacks can only be clear in the application layer which needs to be analyzed to detect this kind of intrusion.

6. High-speed packets capture technology

High-Speed packet capture can reduce resource consumption and improve the detection speed in the network intrusion detection system.

7. Efficient pattern matching algorithm

As intrusions assume more diverse and complex patterns, there is a need to store more complex models in the rule base. It is necessary to develop and use efficient pattern matching algorithms as the complexity of the intrusion model definition is increasing.

8. Test and evaluation of IDS

The development of common IDS evaluation and testing platforms that are necessary for the enhancement of the application is another attractive area of research.

9. Standardization of IDS

No formal international IDS standard so far exists and it is not conducive to develop and apply an IDS.

10. The IDS-security components interaction

The IDS can be combined with other security components either by integration or by cascaded connection.

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

The IDS has its own security issues as well and studies should be conducted on the protection of the IDS itself against network attacks.

2.3 Feature Selection

Feature selection can be defined as a given set of subsets or the selection of subsets that can offer the best classification performance using some classification algorithms. Feature selection can not only minimize the cost of recognition through the reduction of the number of features but can offer a better classification accuracy. From the definitions, it could be seen that evaluation criteria are necessary for any given dataset or learning algorithm. In general, there are four parts of any feature selection algorithm as shown in Figure 2-4. These are the generation, evaluation, stopping and validation criteria. The process of feature selection can be regarded as the removal of redundant features from a given dataset which has little correlation with the class labels. The redundant features may have a strong relationship with the selected features, and therefore, contributes nothing to the classification process(Lin *et al.*, 2012; Jovi, Brki and Bogunovi, 2015).

The generation of the candidate subset involves the search for the feature subsets, and the generated subsets will be employed as input for the evaluation function. The feature selection algorithms initiates with the selection of the initial subset, while the subset generation initiation is divided into three stages: (1) an empty initial subset: while searching, the algorithm adds the individual features to the individual subsets one after the other (known as forwarding search); (2) when the initial subsets are the same as the feature sets of a given dataset: here, the irrelevant or redundant features have been excluded from the initial subset stepwise during the search process (known as a backward search); (3) the initial subset is randomly generated and the feature is added or deleted during the search process one by one(Chandrashekar and Sahin, 2014).

The merits of the individual subsets obtained by the search are evaluated using the evaluation function by comparing the value under evaluation with the previously stored best optimal value. If the observed value is higher, there will be a replacement of the primary candidate subset. The infinite loop state during feature search is avoided using the appropriate termination conditions. A factor that affects the selection of a termination condition is the applied subset search and evaluation function. The speed of finding the best feature during feature selection can be improved by the best feature subset search strategy. There could be a better evaluation function when the classification power of the selected feature subset is higher and can improve the performance of the algorithmic. The strategy for termination condition can be the feature number achieving the specified threshold or a number of search iterations for achieving the specified threshold. The evaluation function-based termination criteria can be the finding of the optimal solution or not obtaining a higher evaluation value by increasing or decreasing the number of feature subsets(Arguello, 2015).

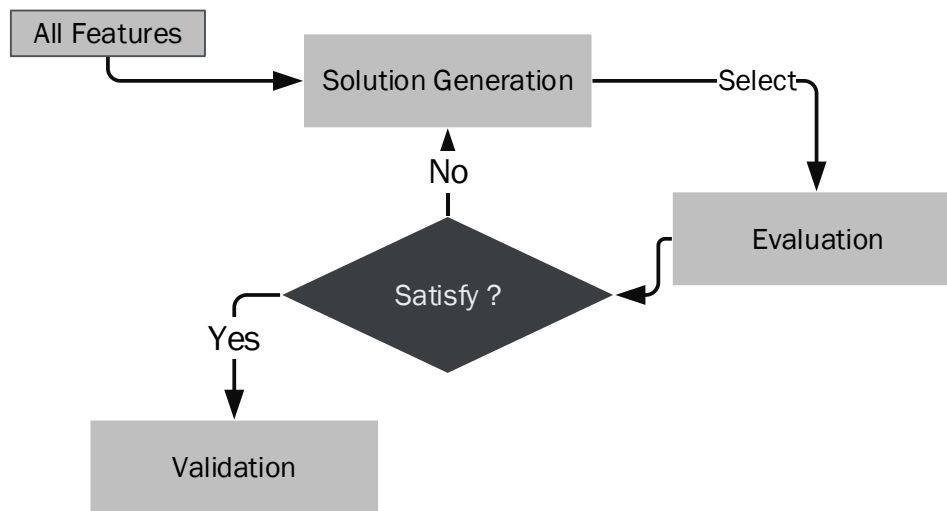


Figure 2-4 Feature Selection Process

The effectiveness of the classification performance during feature selection is validated during the validation process. Though it is not an aspect of the feature selection process, it is an important process for practical application. Validation usually involves the training and testing of feature subsets in the classifier and comparing the prediction results with the results of the original dataset or that of other feature selection results. The comparison may be in terms of classification accuracy or computational complexity(Arguello, 2015).

2.4 Feature Selection Evaluation Measures

There are four major feature selection methods, which are the wrapper method, the hybrid methods, the filter methods, and the embedded methods, as shown in Figure 2-5. Feature selection in the embedded model is incorporated into the training process. Some decision tree algorithms such as ID3, Breiman's CART or C4.5 algorithm selects the best classification features in each node network and then, split the sub-space based on the selected feature. This process is repeated until the termination criterion is met. The embedded method tries to establish the optimal feature subsets during the model building by depending directly on the nature of the employed classification method. Generally, the embedded method has several

advantages in terms of variable and model interaction, being less demanding computationally compared to the wrapper methods, and capturing accurately the dependencies between the variables. These techniques are, however, conceptually more complex and any algorithmic modification may reduce the performance (Chandrashekar and Sahin, 2014; Lee and Kim, 2015).

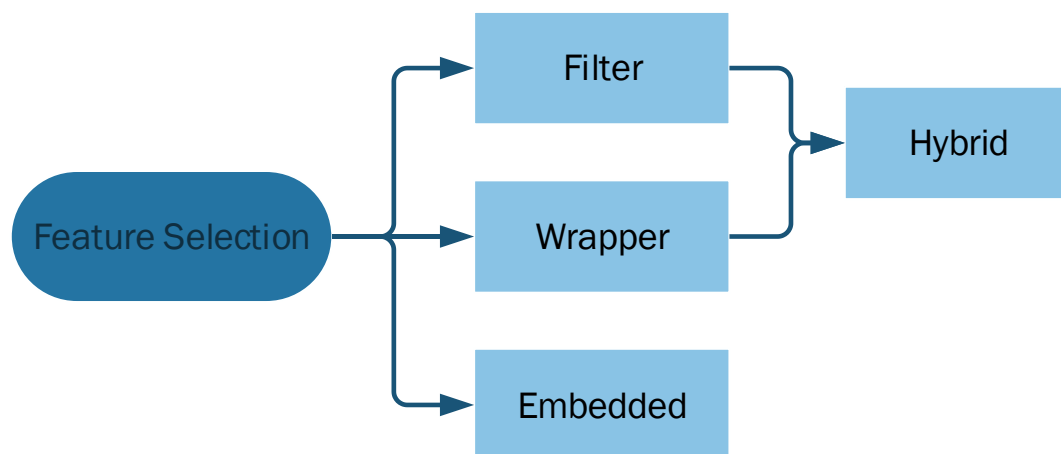


Figure 2-5 Types of feature selection evaluation measure

The filter method evaluation criteria do not depend on the learning algorithms and can feature subsets from the original space based on the given evaluation criteria (Figure 2-6). The evaluation criteria depend on the datasets and filter methods to select a feature or subset that will achieve the best performance in relation to the objective function. The selected feature or subset is generally considered to have higher learning accuracy. Many filter evaluation methods such as inconsistency, correlation, information gain exist. The filter methods have low computational complexity, strong versatility, suitable for large-scale data, and high efficiency.

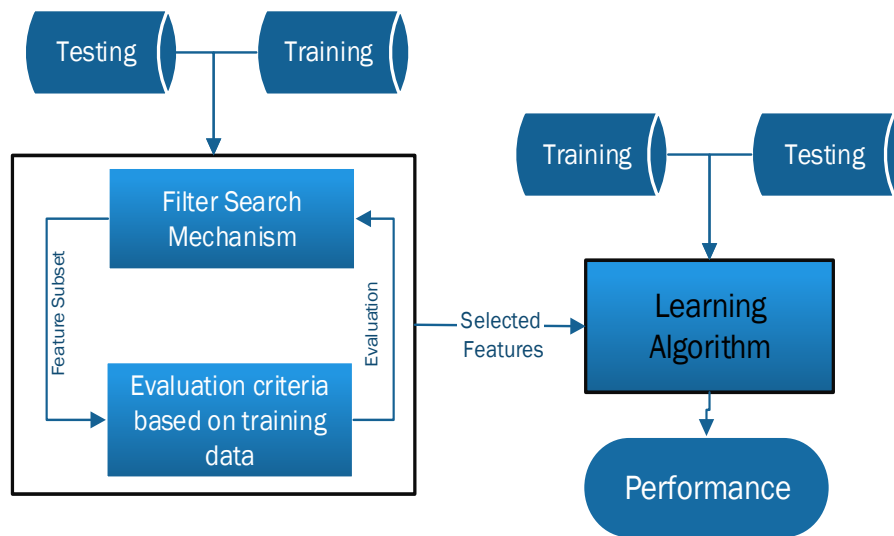


Figure 2-6 Filter-based feature selection

The wrapper methods shown in Figure 2-7 were first suggested by John in 1994. The wrapper methods optimize a classifier as a part of the selection process and select the features with high prediction performance. The selected feature subsets are varied depending on the learning algorithms. The best evaluation criteria used on the selected feature subset is the performance of the learning algorithm. The wrapper methods have no limitation as to the embedded, KNN, Bayesian network, and SVM could be used for the wrapper methods. Generally, the wrapper methods could achieve better subsets than the filter methods, but they require long processing times and have low adaptability, as well as having the need to be trained for different learning algorithms(Guyon and Elisseeff, 2003; Tang, Alelyani and Liu, 2014).

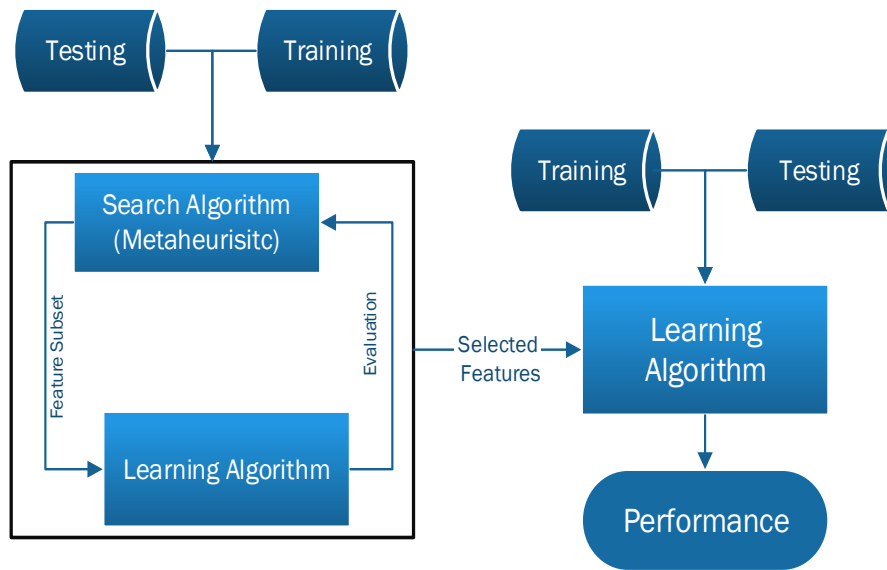


Figure 2-7 Wrapper-based feature selection

The hybrid method is a combination of the filter and wrapper methods and has the advantage of both methods. There are two steps in the hybrid mechanism: at first, the individual features are preprocessed by the filter method to remove the irrelevant features and reduce the size of the dataset; then, the hybrid method selects the features using the wrapper method before evaluating the selected subsets using the classification learning algorithm(Tang, Alelyani and Liu, 2014).

2.5 FEATURE SELECTION PROBLEM IN IDS

The anomaly intrusion detection can be considered a classification problem that can be solved using the machine learning theory. The process of knowledge discovery as proposed by Richard is shown in Figure 2.8 and from the figure, it is evident that feature selection is a data preprocessing task before using a machine learning training algorithm. The machine learning algorithms can help in the creation of the model to differentiate normal and attack patterns.

Intrusion detection can be classified into two types based on the class labels, which are the two-class or multiple classes. A two-class problem considers all attacks as anomaly instances while a multiple class problem considers all attacks as different labels. The datasets for intrusion detection usually have a large number of features and instances and some of these features are redundant. Therefore, feature selection can be applied to this kind of classification domain (Amiri *et al.*, 2011; Davis and Clark, 2011).

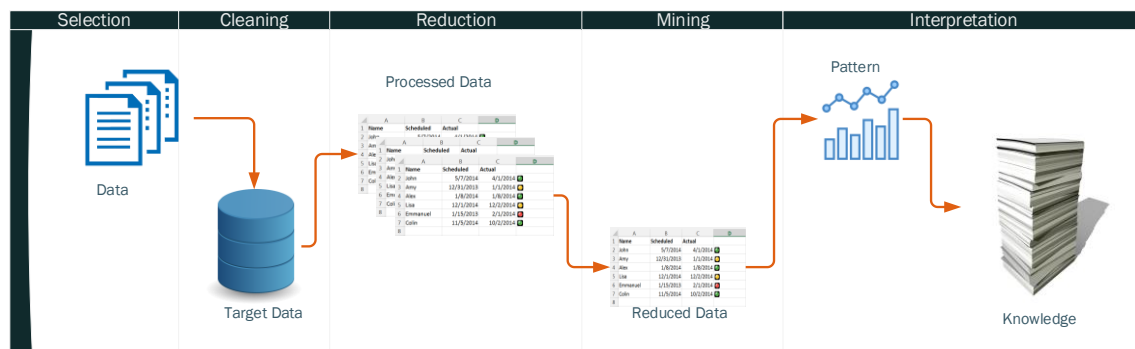


Figure 2-8 Process of Knowledge discovery

There are two learning methods as earlier stated, one is the supervised methods which are based on classifiers such as Bayesian, ID3, JRip, PART and C4.5 (Amor, Benferhat and Elouedi, 2004; Chen *et al.*, 2019), support vector machine and artificial neural network (Mukkamala, Janoski and Sung, 2002) and IBK algorithms (Liao and Vemuri, 2002), the other is the unsupervised methods which are based on the clustering methods such as Fuzzy C Means, Density-based Clustering (Wang *et al.*, 2010), Sub-Space Clustering (SSC) (Casas, Mazel and Owezarski, 2011), and Evidence Accumulation Clustering (EAC) techniques (Casas, Mazel and Owezarski, 2012). A major advantage of unsupervised methods is their ability to detect unknown attacks (Bharti, Shukla and Jain, 2010; Mafarja and Mirjalili, 2018; Taher, Mohammed Yasin Jisan and Rahman, 2019).

As mentioned above, the filter methods do not depend on the learning algorithms and present better performance when dealing with a large dataset. The filter methods evaluate the importance of features through their inherent characteristics such as rough set, distance function, mutual information, statistical correlation coefficient, and independent component analysis(Tang, Alelyani and Liu, 2014).

CHAPTER THREE

DESIGN AND IMPLEMENTATION

3.1 INTRODUCTION

The proposed feature selection algorithm is explained in detail in this chapter, mainly focuses on explaining the hybrid algorithm (GA-FA), and applying it as a wrapper feature selection algorithm for the IDS. The GA-FA uses NB Classifier for calculating the fitness function. There are three sections to be described in this chapter; the first part explains in detail the dataset used in this research. The second part illustrates the general idea of the proposed system with a block diagram, while the third part explains in detail the methodology to be applied to create such a system, and how GA-FA uses the NB classifier to calculate the fitness function for feature selection.

3.2 INTRUSION DETECTION SYSTEM (NSL-KDD)

In IDS research, the KDD Cup 1999 is the commonest data set used. This data contains about 125.000 connection records and each record consists of 41 features ('KDD Cup 1999 Dataset', no date). This data has been statistically analyzed and presented (Tavallaei *et al.*, 2009). In the KDD data set, there are four major categories of attacks; they are:

- **Denial of Service (DoS):**

D.o.S is a form of attack in which the intruder has access to the computing accessories and makes the system too clustered or busy to consider genuine requests, thereby, denying access to legitimate users.

- **Surveillance and Other Probing:**

Probing is a situation where an attacker can scan the network and identify system vulnerabilities to exploit based on the gained information.

- **Unauthorized Access from a Remote Machine (R2L):**

A remote to user (R2L) attack is a situation where a packet is sent by an attacker to a network machine, then, exploit the weakness of that machine to gain unlawful access to the network as a regular user.

- **Unauthorized Access to Local Super User (U2R):**

User to root is a situation where an attacker can access a network as a regular user and then, exploit the network susceptibility to getting root access.

Many ML and pattern classification algorithms have been used to solve intrusion detection problems based on the KDD dataset, but have all failed to detect most of the remote-to-local and user-to-root attacks. The limitations of the KDD data set has been identified (Sabhnani and Serpen, 2004) and suggested not to be used in training pattern recognition or ML algorithms for misuse detection of these two attack categories.

NSL-KDD data set, it has been reported that the KDD dataset has many problems; for example, it contains several redundant features, and the difficulty level of the different records and the percentage of records in the original KDD dataset are not inversely proportional. These deficits result in a poor evaluation of different proposed ID techniques. The NSL-KDD dataset was proposed to overcome some of these inherent problems of the KDD Cup 1999 data set. The proposed new dataset consists of selected records of the complete KDD dataset (Tavallae *et al.*, 2009). Table 3-1 showed the NSL-KDD data variables, while Table 3-2 showed the distribution of attack records per attack category (Kang and Kim, 2016). The following are some of the advantages of the NSL-KDD over the original KDD dataset (Tavallae *et al.*, 2009):

- 1) Redundant records are excluded in the training set. Thus, there is no bias towards more frequent records.
- 2) In the original KDD data set, the number of records selected from each group level and the percentage of records is inversely related.
- 3) If there is a sound number of records in the training and testing portions, experiments on the whole set can be economically tested without the necessity for a random sample at a reduced scale.

Table 3-1 The features of the NSL-KDD data set

F	Name	Type	Max
1.	duration	Numeric	54,451
2.	protocol_type	Symbolic	2
3.	service	Symbolic	64
4.	flag	Symbolic	10
5.	src_bytes	Numeric	89,581,520
6.	dst_bytes	Numeric	7,028,652
7.	land	Boolean	1
8.	wrong_fragment	Numeric	3
9.	urgent	Numeric	3
10.	hot	Numeric	101
11.	num_failed_logins	Numeric	4
12.	logged_in	Boolean	1
13.	num_compromised	Numeric	7479
14.	root_shell	Numeric	1
15.	su_attempted	Numeric	2
16.	num_root	Numeric	7468
17.	num_file_creations	Numeric	100
18.	num_shells	Numeric	2
19.	num_access_files	Numeric	9
20.	num_outbound_cmds	Numeric	0
21.	is_host_login	Boolean	1
22.	is_guest_login	Boolean	1
23.	count	Numeric	511
24.	srv_count	Numeric	511
25.	serror_rate	Numeric	1.0
26.	srv_serror_rate	Numeric	1.0
27.	rerror_rate	Numeric	1.0
28.	srv_rerror_rate	Numeric	1.0
29.	same_srv_rate	Numeric	1.0
30.	diff_srv_rate	Numeric	1.0
31.	srv_diff_host_rate	Numeric	1.0
32.	dst_host_count	Numeric	255
33.	dst_host_srv_count	Numeric	255
34.	dst_host_same_srv_rate	Numeric	1.0
35.	dst_host_diff_srv_rate	Numeric	1.0
36.	dst_host_same_src_port_rate	Numeric	1.0
37.	dst_host_srv_diff_host_rate	Numeric	1.0
38.	dst_host_serror_rate	Numeric	1.0
39.	dst_host_srv_serror_rate	Numeric	1.0
40.	dst_host_rerror_rate	Numeric	1.0
41.	dst_host_srv_rerror_rate	Numeric	1.0

All min=0

Table 3-2 Distribution of attack records per NSL-KDD attack category

Attack Category	Attack Name	No. of Records
DoS	Back	956
	Land	18
	Neptune	41214
	Pod	201
	Smurf	2646
	Teardrop	892
		45927
Probe	Satan	3633
	Ipsweep	3599
	Nmap	1493
	Portsweep	2931
		995
R2L	Guess Password	53
	Ftp_write	8
	Imap	11
	Phf	4
	Multihop	7
	Warezmater	20
	Warezcinet	890
	Spy	2
		995
	Buffer_overflow	30
	Loadmodule	9
	Rootkit	10
	Perl	3
Normal		67343
U2R		52
Total		125973

3.3 THE PROPOSED INTRUSION DETECTION SYSTEM

The ultimate aim of this work is to propose a hybrid wrapper feature selection algorithm, which has the ability to select the most relevant features from the NSL-KDD dataset, which enhances the detection rate of the Network Intrusion Detection System. In addition, the selection process reduces the size of the dataset, which leads to a decrease in the required memory space and less search space for classification and high accuracy models.

The proposed IDS system consists of three main stages, Preprocessing stage, Feature Selection Stage, Evaluation Stage. Figure 3.1 shows the general block diagram of the proposed system.

Stage 1: Preprocessing

The whole dataset is preprocessed in this stage. It consists of two steps, scaling and normalization. In the scaling step, the dataset is converted from string representation into a numerical representation. For example, the class label in the dataset contains two different categories ‘Normal’ and ‘Attack’, after implementing this step this label is changed into ‘1’ and ‘0’. Where ‘1’ means the normal case, while ‘0’ means attack.

The second step is normalization. decreases the differences in the ranges between the features. In this work, we have used the Min-Max normalization method, as follows:

$$F_i = \frac{(F_i - Min_i)}{(Max_i - Min_i)} \quad (3.1)$$

Where F_i represents the current feature needs to be normalized, Min_i and Max_i represent the minimum and the maximum value for that feature respectively.

Stage 2: Feature Selection

This stage is responsible for choosing the most relevant feature from the preprocessed dataset. It consists of two steps, the Genetic Algorithm – Firefly Algorithm (GA-FA) and Naïve Bayesian Classifier. In the first step, the algorithm will generate N fireflies (i.e. Swarm size), all these searching agents are evaluated by using a naïve Bayesian classifier which represents the fitness function in this work. The next section explains the main steps of the hybrid algorithms in detail.

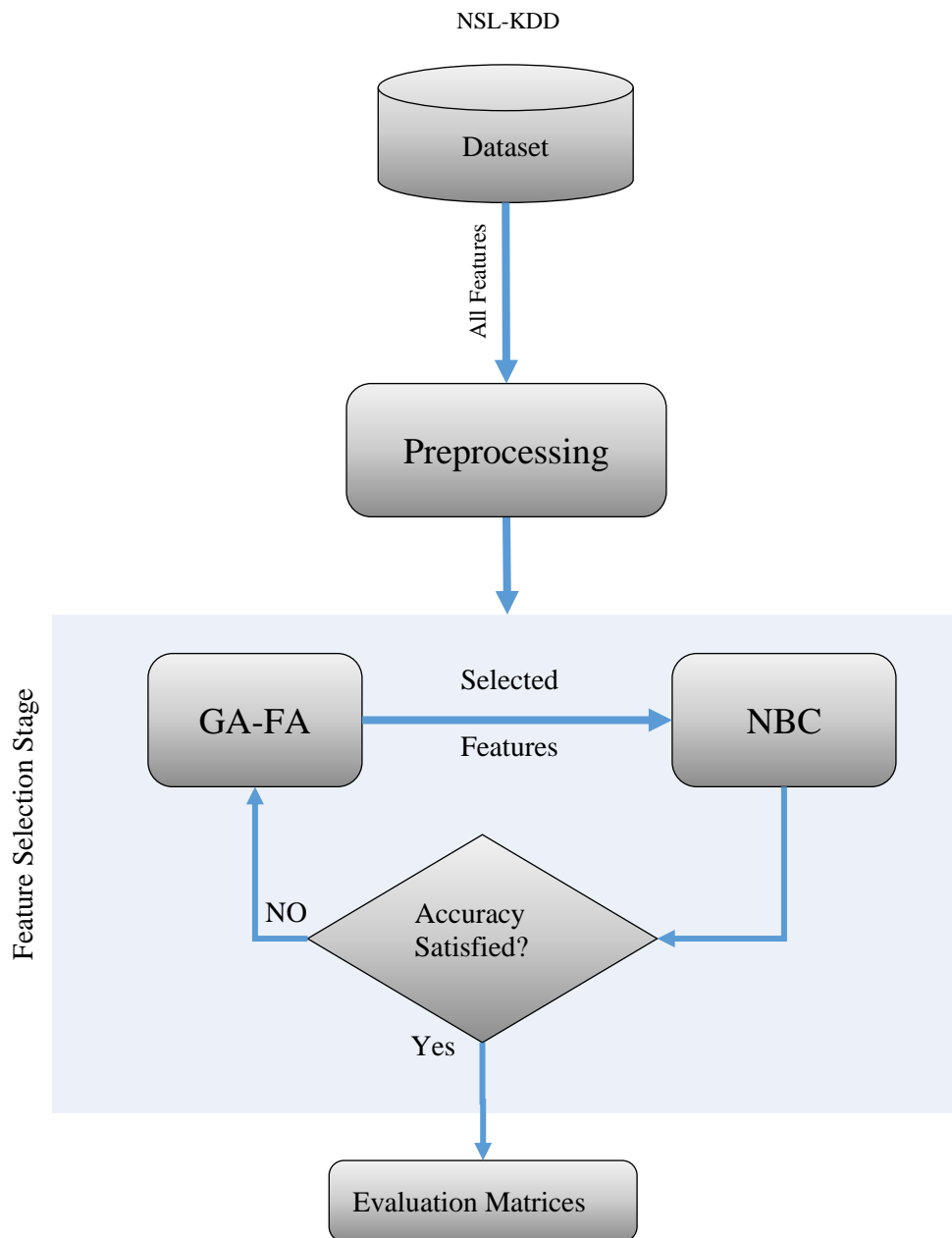


Figure 3-1 Block diagram of the proposed system

Stage 3: Evaluation Matrices

The evaluation process estimates the validity and accuracy of these constructed models which are obtained by the following classification counters:

1. True Positive (TP): Represents the number of correctly classified samples for the positive class.
2. False Positive (FP): represents the number of incorrectly classified samples for the positive class.
3. True Negative (TN): represents the number of correctly classified samples for the negative class.
4. False Negative (FN): represents the number of incorrectly classified samples for the negative class.

There are four evaluation measures that are used to evaluate the results of classifiers which depend on four classification counters as follows:

Accuracy evaluates the classifier's effectiveness through its correct predictions percentage.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (3.2)$$

Error rate evaluates the classifier through its incorrect predictions percentage.

$$\text{Error rate} = \frac{\text{FP} + \text{FN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (3.3)$$

The precision measure estimates the probability of correct positive prediction.

3.4 THE PROPOSED FEATURE SELECTION ALGORITHM

3.4.1 Firefly Algorithm

The FA was developed by Yang (Yang, 2008) as a stochastic nature-inspired optimization technique that was inspired by the light flashing and mating pattern of fireflies. Globally, there are more than 2000 species of fireflies, and most of these fireflies' short and rhythmic flash patterns radiate (Yang, 2010; Ghorbani *et al.*, 2017). The basic action of these flashes includes the attraction of partners for communication (mating), the attraction of potential prey, and the provision of a mechanism of warning. There are two elements that visualize most fireflies within a given distance only; the first one is the light intensity at a given distance r from the light source. This condition follows the law of inverse square which suggests a light intensity I reduce as the distance rises $I \propto \frac{1}{r^2}$, i.e., the inverse

proportionality of light intensity to the square of the distance. The other factor is the light absorption in air. The light absorption in the air reduces the light intensity as the distance increases. In the artificial firefly, these rules are idealized as discussed in the next part. The general flowchart of FA is given in Figure 3-2, the behavior of artificial fireflies is governed by the following 3 idealized rules as listed by Yang:

- Regardless of sex, all fireflies can be attracted to each other because it is unisex.
- The attractiveness degree of a particular firefly is depended on its luminosity; thus, brighter fireflies usually attract fireflies with lesser luminosity. The decreases of attractiveness effect with an increase in the distance among the fireflies
- The firefly luminosity is a function of the objective function (OF). The luminosity of the light produced by a firefly is proportional to the value of the OF for a maximization problem.

The attractiveness between two fireflies i and j with another one which has more brighter than the first j is determined by (Yang, 2008, 2014):

$$X_i = (X_j - X_i) \beta_0 e^{-\gamma r^2} + \alpha (rand - 0.5) \quad (3.4)$$

where j and i are the randomization and attraction. α is the randomization parameter, and the $rand$ is a Random distribution uniformly number in the range of $[0, 1]$. Therefore, the expression $(rand - 0.5)$ is in the range of $[-0.5, 0.5]$ to give room for both negative and positive changes. β_0 is usually fixed to 1 and $\alpha \in [0; 1]$. Parameter α is the noise within the environment which can affect the transmission of light. This parameter can be selected in the artificial algorithm to allow disparity in the solution and offer more solutions. The parameter is also extendable to a normal distribution with the mean 0 and the variance 1; $N(0; 1)$. This will permit changes within the environment noise rate. γ is the changes in attractiveness; the value of this parameter is important to identify the speed of proximity and behavior of FA. It varies from 0.01 to 100 most uses. The rang from f_i to f_j is denoted as f_{ij} and described by Equation (3.5) (Yang, 2008, 2014).

$$f_{ij} = \| X_i - X_j \| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad (3.5)$$

In which X_i = the firefly i location . notice it's in that updating equation $\beta_0 e_{ij}^{-\gamma r^2}$, the parameter of attractiveness is utilized as an approximation of the intensity loss of light due to rang as already described in the idealization laws. It can be designed with decreasing function monotonically. Similarly, the random term in the equation can be used to model the effect of dust and the environment on the intensity of light.

The FA attributes can be idealized as follows:

- A swarm intelligent technique with the benefits of swarm optimization.
- An algorithm with the capability of easily treatment multi-model problems due to its automatical population partitioning, where the vision scope of each firefly is restricted to allow the of sub-swarms formation within the solution space.
- The approaching speed of the FA can be enhanced by tuning its randomness and attraction parameters throughout the iteration process.

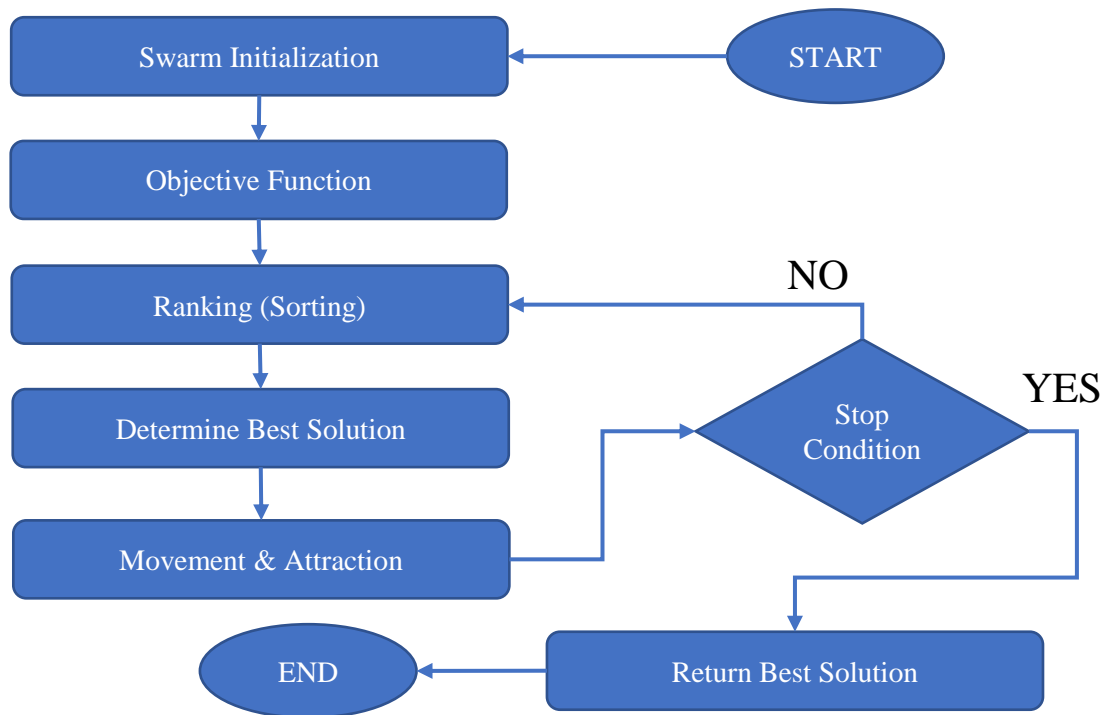


Figure 3-2 Flowchart of standard FA

3.4.2 The Proposed FA

The firefly algorithm (FA) has been used for solving different types of optimization problems in the literature. Although the original FA has attained good results, it has a weakness with the balancing between the global search and local search. The local search ability is executed more than the global search which leads to increase the chances for the algorithm to get trapped in the local optima. Therefore, the main contribution of this thesis is to enhance the global search ability of FA using an evolutionary operator implemented in the Genetic Algorithm (GA) which is the mutation operator.

In the original version of FA, the best firefly – or best solution – does not move to any position, instead of that, all of the other fireflies in the swarm move towards it. If the algorithm does not find a better position for a number of iterations, FA will not enhance. The crossover operator helps FA by moving the best firefly in the swarm towards a new random position. In other words, the best solution is updated using the crossover operator randomly, which means it may explore a better position and as a result, the algorithm will be enhanced. The general flowchart of GA-FA is given in the figure below.

The proposed feature selection consists of the following main steps:

1- Swarm Initialization

Each firefly in the swarm is initialized randomly in a continuous domain using a chaotic logistic map (see eq. 3.6). For solving the feature selection problem, each firefly should be encoded in binary representation. Therefore, the continuous values are converted using sigmoid function into 0s and 1s, where 1 represents the selected features, and 0 represents non selected features (see eq. 3.7) (Mafarja *et al.*, 2017). The general structure for each firefly after the binary encoding is given in Figure 3-3 below.

$$X_{i+1} = X_i \mu (1 - X_i) \quad (3.6)$$

$$F_i = \begin{cases} 0 & \frac{1}{1+e^{X_i}} < \text{Random}(0,1) \\ 1 & \text{Otherwise} \end{cases} \quad (3.7)$$

Where X_i denotes the continuous value of the firefly, μ represents the mutation value or the control parameter for the logistic map, and F_i represents the binary value of fireflies.

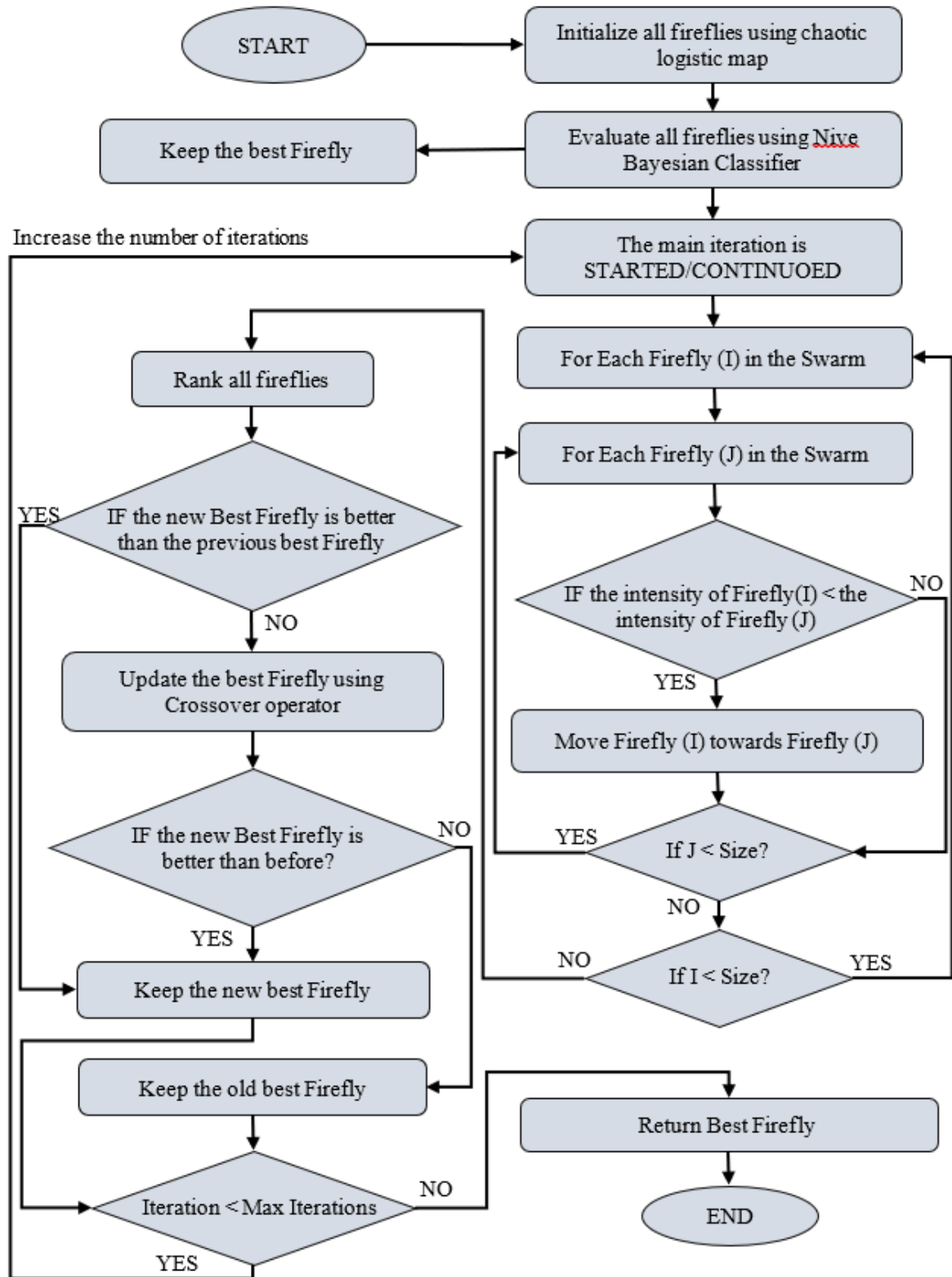


Figure 3-3 Flowchart of GA-FA

Firefly	Features Set (n)						Fitness(m)
	F_1	F_2	F_3	F_4	F_n	
Firefly 1	0	1	1	0	...	1	0.854
Firefly 2	1	1	0	1	...	0	0.254
Firefly 3	1	0	0	0	...	0	0.124
Firefly N	0	0	1	1	...	1	0.475

Figure 3-4 The structure for each firefly

2- Objective or Fitness Function

A fitness function is needed to guide the search by assigning to any tentative solution a quality value. In the objective function, both accuracy and number of features are used to evaluate the solutions, based on the following equation(3.8):

$$\min f(x) = (100 - Accuracy) + (a * \#features) \quad (3.8)$$

Where $\#features$ represents the number of selected features, and a is a constant value used for normalizing the value of $\#features$. Therefore if the number of features in the subset is high (with regards to the total numbers of features in the original dataset), then the fitness function promotes the reduction of features. Otherwise, if the number of features in the subset is small, then this fitness function promotes the improvement in accuracy.

The accuracy is obtained using naïve bayesian classifier (NBC), which can be calculated based on the probability of the features. The predicted class is calculated based on the output of the maximum probabilities for all possible values as follows:

$$y = \text{Max}_y P(y) \prod_{i=1}^n P(x_i, y) \quad (3.9)$$

Where y represents the class for the samples, while x represents the values for the input values of the features. The function $P(x)$ represents the probability of the feature.

3- The Position Updating of Fireflies

Each firefly in the swarm is moved towards another firefly with higher intensity via eq (3.4), meaning that the position of each firefly is updated in the continuous form. Then, these fireflies should be re-converted to binary representation using the method in step 1.

4- Crossover Operator for Best Firefly

As explained at the beginning of this subsection, the best firefly always stays in the same position without any movement. This leads to slow the searching process and may increase the chances of falling the local optima. In the enhanced version, The crossover operator is imported from the genetic algorithm to update the position of the best firefly, by exchanging random variables or features between the best firefly and the second-best firefly. Figure 3-5 illustrates this operator. The procedures for the crossover operator between these two fireflies are executed as follows:

- A. Set the number of Features to be swapped between these fireflies randomly in pre-identified range) as *Swap*.
- B. Select a random feature in the best firefly $X_{r_1}^{Best}$ and swap it with a random feature in the second-best firefly $X_{r_2}^{Second}$. Where r_1 and r_2 represent the random feature in the best and the second fireflies respectively.
- C. If the number of swapped features less than the *Swap*, then go to Step B, otherwise go to Step D.
- D. Evaluate the new best firefly via eq (3.8), if the new fitness value is better than the previous value, then Keep it, otherwise, keep the previous best firefly.

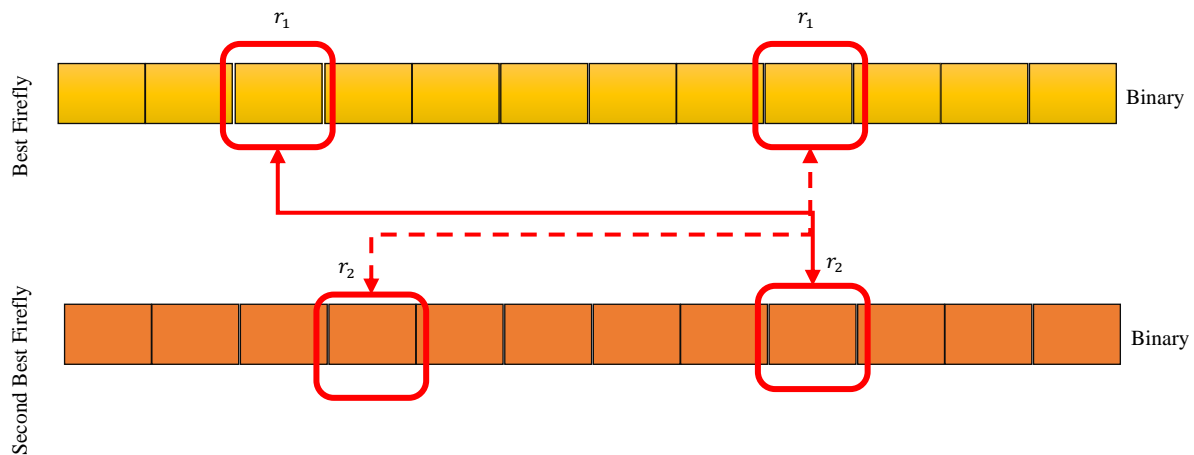


Figure 3-5 A graphical illustration for Crossover operator

It is important to mention that the procedures of the crossover operator between the two fireflies swapped both types of variables (i.e., real and binary) as illustrated in figure(3.5).

3.5 SUMMARY

In the first and second sections, the well-known dataset (NSL) and, the proposed system, in general, have been described. The third section explained the original firefly algorithm and explained the proposed feature selection algorithm with the crossover operator. This section showed that the original firefly the main drawback of FA, which is the lack of global searchability. Then presented the solution by explaining how the crossover operator solved it.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Introduction

The main contribution of this study is the designing of an enhanced version of the firefly algorithm by using a crossover operator for selecting the most relevant subset of features in the dataset of IDS. The proposed algorithm has been explained in chapter 3 in detail. This chapter presents the results and discussion of the proposed system. The chapter is divided into two main parts. The first part illustrates the experimental setup, while the second part presents the attained results.

4.2 Experimental Settings

The main code of the proposed GA-FA and naïve Bayesian classifier have been developed by using visual C#.net version 6.0 – Visual Studio 2017 community version. The developed program has been implemented in the environment with the following specification: Operating System is Windows 10 with 64-bit architecture, CPU Intel 2.4 GHz, and RAM 8GB.

As mentioned in chapter 3, the dataset used in this thesis is the NSL-KDD dataset. The original version of NSL-KDD consists of five classes (Normal, DDOS, R2L, U2L, and Prop), 41 features, and around 125000 samples. In this work, we have converted the dataset into binary classes (Attack and Normal),. In order to evaluate the proposed algorithm, several experiments have been done.

Swap, to test the effect of the swapped variables on the searching process, or especially on the global search ability of the GA-FA algorithm, four cases were examined (5, 10, 15, 20).

4.3 Results and Discussion

4.3.1 The Effect of Swarm Size and Number of Iterations

In this section, the proposed algorithm has been tested based on two main characteristics. These experiments were different in terms of the number of iterations, and the number of swarm size, as follows:

- 1- Swarm Size, to test the effect of the food sources or swarm size on the searching process, four cases were used in the experiments (10, 20, 30, 40).
- 2- Number of Iterations, to test the effect of the iterations on the searching process, three cases were examined, (100, 250) the number of iterations.

As mentioned in chapter 3, the GA-FA algorithm is initialized randomly, each run time it started from different positions, which lead to different results. Therefore, all experiments above have been implemented 15 run times, the best, the worst and the mean accuracy are measured. In addition, the best, the worst and the mean of the selected features are measured as well. The *Swap* value in all of these experiments is equal to 10.

Tables below, present all scenarios for the above-mentioned experiments. Each table presents the original accuracy obtained by naïve Bayesian classifier, and the predicted accuracy after applying the proposed algorithm, precision, recall, the number of selected features, and the number of removed features. The number of all scenarios is equal to 8. Table 4-1 presents the parameter settings for the proposed algorithm.

Table 4-1 Parameter Settings

N	Parameter	Symbol	Value
1	Logistic Map Initial value	X_0	Random [0,1]
2	Initial Attractiveness	B_0	0
3	Randomization Factor	a	0.2

4	Gamma	γ	1.0
5	Delta	δ	0.96

Scenario 1:

Swarm Size = 10

No. of Iterations = 100

The results of the 15 run times are presented in the following table

Table 4-2 The results of 15 run times for scenario 1

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.03333	0.91598	0.99617	16	25
2.	89.6	94.10000	0.90116	0.99617	14	27
3.	89.6	94.66667	0.90116	0.99489	15	26
4.	89.6	94.56667	0.91333	0.98978	14	27
5.	89.6	95.40000	0.92445	0.99297	15	26
6.	89.6	94.86667	0.91622	0.99233	16	25
7.	89.6	94.56667	0.91187	0.99169	12	29
8.	89.6	93.76667	0.90012	0.99042	18	23
9.	89.6	94.96667	0.91785	0.99233	13	28
10.	89.6	93.53333	0.89785	0.98850	17	24
11.	89.6	95.43333	0.92754	0.98978	17	24
12.	89.6	94.93333	0.91437	0.99617	16	25
13.	89.6	94.46667	0.90551	0.99808	21	20
14.	89.6	94.33333	0.90861	0.99105	16	25
15.	89.6	94.30000	0.90429	0.99617	18	23

Scenario 2:

Swarm Size = 10

No. of Iterations = 250

Table 4-3 results of 15 run times for scenario 2

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	94.90000	0.93154	0.97380	11	30
2.	89.6	94.43333	0.90545	0.99744	17	24
3.	89.6	94.03333	0.90197	0.99361	19	22
4.	89.6	95.56667	0.93185	0.98722	14	27
5.	89.6	94.76667	0.91315	0.99425	18	23
6.	89.6	94.30000	0.90336	0.99744	20	21
7.	89.6	95.53333	0.92973	0.98914	12	29
8.	89.6	95.26667	0.92026	0.99553	12	29
9.	89.6	95.26667	0.91828	0.99808	20	21
10.	89.6	95.60000	0.92827	0.99233	16	25
11.	89.6	95.80000	0.93007	0.99425	13	28
12.	89.6	95.43333	0.92703	0.99042	16	25
13.	89.6	95.60000	0.92573	0.99553	18	23

14.	89.6	94.93333	0.91583	0.99425	19	22
15.	89.6	95.36667	0.92040	0.99744	20	21

Scenario 3:

Swarm Size = 20

No. of Iterations = 100

Table 4-4The results of 15 run times for scenario 3

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.00000	0.91889	0.99169	15	26
2.	89.6	94.60000	0.90761	0.99808	17	24
3.	89.6	94.00000	0.90145	0.99361	19	22
4.	89.6	95.13333	0.91760	0.99617	16	25
5.	89.6	95.50000	0.92158	0.99872	19	22
6.	89.6	95.53333	0.92665	0.99297	13	29
7.	89.6	95.60000	0.92827	0.99233	18	23
8.	89.6	94.50000	0.91274	0.98914	18	23
9.	89.6	94.66667	0.92091	0.98211	16	25
10.	89.6	95.00000	0.91593	0.99553	16	25
11.	89.6	94.53333	0.90703	0.99744	15	26
12.	89.6	95.06667	0.91554	0.99744	18	23
13.	89.6	94.60000	0.91047	0.99425	18	23
14.	89.6	94.46667	0.90457	0.99936	19	22
15.	89.6	95.30000	0.91981	0.99681	17	24

Scenario 4:

Swarm Size = 20

No. of Iterations = 250

Table 4-5The results of 15 run times for scenario 4

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.26667	0.92326	0.99169	15	26
2.	89.6	95.50000	0.93176	0.98594	13	28
3.	89.6	94.90000	0.91432	0.99553	16	25
4.	89.6	94.53333	0.90845	0.99553	16	25
5.	89.6	95.26667	0.92681	0.98722	14	27
6.	89.6	95.30000	0.92180	0.99425	19	22
7.	89.6	94.66667	0.91202	0.99361	15	26
8.	89.6	95.60000	0.92827	0.99233	11	30
9.	89.6	95.10000	0.91854	0.99425	17	24
10.	89.6	95.06667	0.92349	0.98722	14	27
11.	89.6	96.03333	0.94247	0.98403	14	27
12.	89.6	96.00000	0.93814	0.98850	11	30

13.	89.6	95.13333	0.91908	0.99425	14	27
14.	89.6	94.93333	0.91437	0.99617	18	23
15.	89.6	95.06667	0.91652	0.99617	17	24

Scenario 5:

Swarm Size = 30

No. of Iterations = 100

Table 4-6 The results of 15 run times for scenario 5

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	96.30000	0.94711	0.98403	13	28
2.	89.6	94.86667	0.91233	0.99744	20	21
3.	89.6	95.30000	0.91833	0.99872	18	23
4.	89.6	94.96667	0.91539	0.99553	17	24
5.	89.6	95.30000	0.91833	0.99872	12	29
6.	89.6	94.60000	0.90999	0.99489	16	25
7.	89.6	95.33333	0.92085	0.99617	17	24
8.	89.6	94.76667	0.91218	0.99553	17	24
9.	89.6	94.86667	0.91427	0.99489	14	27
10.	89.6	95.56667	0.92619	0.99425	14	27
11.	89.6	94.83333	0.91228	0.99681	18	23
12.	89.6	95.00000	0.91642	0.99489	19	22
13.	89.6	94.76667	0.91855	0.98722	14	27
14.	89.6	94.90000	0.91874	0.98978	16	25
15.	89.6	95.93333	0.93385	0.99233	15	26

Scenario 6:

Swarm Size = 30

No. of Iterations = 250

Table 4-7 The results of 15 run times for scenario 6

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.36667	0.92695	0.98914	16	25
2.	89.6	95.80000	0.93007	0.99425	15	26
3.	89.6	95.26667	0.92528	0.98914	18	23
4.	89.6	95.73333	0.92641	0.99744	16	25
5.	89.6	95.40000	0.92244	0.99553	15	26
6.	89.6	95.36667	0.92340	0.99361	15	26
7.	89.6	95.20000	0.93244	0.97891	11	30
8.	89.6	95.43333	0.92299	0.99553	20	21
9.	89.6	96.10000	0.93457	0.99489	16	25
10.	89.6	95.33333	0.92385	0.99233	12	29
11.	89.6	95.60000	0.92776	0.99297	17	24
12.	89.6	95.30000	0.92482	0.99042	16	25
13.	89.6	95.73333	0.92742	0.99617	18	23
14.	89.6	96.21000	0.93690	0.99617	16	25

15.	89.6	95.90000	0.93486	0.99042	17	24
-----	------	----------	---------	---------	----	----

Scenario 7:

Swarm Size = 40

No. of Iterations = 100

Table 4-8 The results of 15 run times for scenario 7

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.50000	0.92610	0.99297	15	26
2.	89.6	95.66667	0.94427	0.97444	14	27
3.	89.6	95.60000	0.92930	0.99105	15	26
4.	89.6	95.33333	0.92588	0.98978	16	25
5.	89.6	95.93333	0.93490	0.99105	15	26
6.	89.6	95.56667	0.92721	0.99297	16	25
7.	89.6	95.20000	0.91917	0.99553	17	24
8.	89.6	96.10067	0.90598	0.99744	19	22
9.	89.6	95.56667	0.92568	0.99489	18	23
10.	89.6	94.76667	0.91170	0.99617	18	23
11.	89.6	94.83333	0.91617	0.99169	18	23
12.	89.6	95.93333	0.92972	0.99744	18	23
13.	89.6	95.96667	0.94132	0.98403	12	29
14.	89.6	94.73333	0.91751	0.98786	13	28
15.	89.6	95.63333	0.93297	0.98722	15	26

Scenario 8:

Swarm Size = 40

No. of Iterations = 250

Table 4-9The results of 15 run times for scenario 8

No	Original Accuracy	Predicted Accuracy	Precision	Recall	Selected Features	Removed Features
1.	89.6	95.66667	0.93459	0.98594	11	30
2.	89.6	95.60000	0.92674	0.99425	13	28
3.	89.6	95.40000	0.92395	0.99361	14	27
4.	89.6	95.40000	0.92194	0.99617	17	24
5.	89.6	95.26667	0.92733	0.98658	15	26
6.	89.6	95.70000	0.92994	0.99233	16	25
7.	89.6	96.06667	0.93663	0.99169	14	27
8.	89.6	95.70000	0.92789	0.99489	19	22
9.	89.6	95.33333	0.92385	0.99233	12	29
10.	89.6	95.23333	0.92221	0.99233	15	26
11.	89.6	95.30000	0.91932	0.99744	17	24
12.	89.6	95.23333	0.91774	0.99808	20	21
13.	89.6	96.40667	0.94254	0.98530	14	27
14.	89.6	95.80000	0.92955	0.99489	17	24
15.	89.6	95.73333	0.93101	0.99169	15	26

Table 4-2 below summarizes the results presented above. As can be seen, the results are increased gradually with the increase of the swarm size. Thus, the swarm size effect on accuracy. In addition, the number of iterations also affect the searching process. We can conclude that both swarm size and the number of iterations is the effect on prediction accuracy when they are increased.

Table 4-2 The summarized results for the proposed algorithm

No. of Iterations	Swarm Size	Best Accuracy	Worst Accuracy	Average Accuracy	Average Features
100	10	95.43333	93.53333	94.5	15.8
	20	95.60000	94.00000	94.9	16.9
	30	96.30000	94.60000	95.1	16
	40	96.10067	94.76667	95.36	15,9
250	10	95.80000	94.03333	95.05	16.3
	20	96.00000	94.53333	95.22	14.9
	30	96.21000	95.20000	95.5	15.8
	40	96.40667	95.23333	95.7	15.2



Figure 4-1 The effect of swarm size and iteration number on the accuracy

4.3.2 The Effect of *Swap* Variable

In this section, the effect of the *Swap* variable which controls the number of variables swapped between the best firefly and the second firefly. The results of all experiments for the proposed algorithm are presented in this section. Each table presents the summarized results for different swarm sizes (20, 30, 40, 50, 60) and the number of iterations (50, 100, 250, 500). The *Swap* of Table 4.1 shows the results of 16 experiments when the limit is equal to 5, Table 4.2 shows the results of 16 experiments when the swap is equal to 10, Table 4.3 shows the results of 16 experiments when the swap is equal to 15, Table 4.4 shows the results of 16 experiments when the swap is equal to 20.

Table 4-3 Results of the proposed algorithm for *Swap* = 5

Iterations	Swarm Size	Accuracy			Selected Features		
		Best	Worst	Mean	Best	Worst	Mean
50	20	96.1	92.8	94.82778	12	13	13
	30	96.2	93.4	95.06111	11	16	13.3
	40	96.25	92.8	94.83222	12	13	11.96667
	60	96.3333	93.9	95.33111	13	15	13.16667
100	20	96.06667	94.46667	95.30333	11	15	12.73333
	30	96.3	93.86667	95.52111	12	14	12.73333
	40	96.46667	94.3	95.29667	11	15	12.6
	60	96.6	94.36667	95.73889	10	13	13
250	20	96.13333	94.86667	95.57444	14	8	12.9
	30	96.4	94.46667	95.69778	8	10	12.4
	40	96.46667	94.96667	95.95	13	15	12.96667
	60	96.63333	95.2	95.98778	12	11	12.26667
500	20	96.3	94.96667	95.99333	11	14	12.36667
	30	96.56667	95.66667	96.04111	9	12	11.83333
	40	96.90111	95.16667	96.00111	10	11	11.93333
	60	97	95.46667	96.12889	13	17	11.8

Table 4-4 Results of the proposed algorithm for *Swap* = 10

Iterations	Swarm Size	Accuracy			Selected Features		
		Best	Worst	Mean	Best	Worst	Mean
	20	96.36667	92.66667	94.57111	8	11	12.53333

50	30	96.43333	92.56667	95.12111	12	19	13.43333
	40	96.43333	93.66667	95.01778	14	18	13.8
	60	96.56667	94.4	95.5	12	15	13.1
100	20	96.43333	93.73333	95.33111	11	12	13.4
	30	96.46667	94.36667	95.49556	10	13	12.43333
	40	96.66667	93.66667	95.52444	9	14	12.6
	60	96.73333	94.13333	95.6056	16	17	12.43333
250	20	96.56667	94.83333	95.60444	12	16	13.06667
	30	96.56667	94.16667	95.79111	9	16	11.8
	40	96.66667	94.96667	95.91889	12	18	12.76667
	60	96.76667	95.0667	95.8978	15	11	11.63333
500	20	96.57	95.2	95.96556	11	9	11.63333
	30	96.67	95.36667	96.06	14	18	11.86667
	40	96.76667	95.63333	96.15889	13	10	11.93333
	60	96.83333	95.56667	96.13222	9	15	11.66667

Table 4-5 Results of the proposed algorithm for Swap = 15

Iterations	Swarm Size	Accuracy			Selected Features		
		Best	Worst	Mean	Best	Worst	Mean
50	20	96	92.16662	94.64889	12	15	13.63333
	30	96.36667	92.7	94.90444	14	17	12.33333
	40	96.26667	93.9	95.05444	12	16	12.73333
	60	96.33333	93.26667	95.32778	12	12	12.46667
100	20	96.05	93.7	95.15	13	16	13
	30	96.53333	94.53333	95.46667	14	14	12.9
	40	96.63333	94.06667	95.32111	14	12	12.3333
	60	96.63333	94.43333	95.57111	9	15	12.26667
250	20	96.3	93.8	95.43333	15	13	12.43333
	30	96.7	94.8	95.79889	12	14	12.3
	40	96.68887	95.23333	95.83556	13	12	11.7
	60	96.75333	95.43333	95.93889	11	13	12.06667
500	20	96.53333	94.5	95.85667	12	14	12.76667
	30	96.86667	95.2	95.93111	11	12	12.7
	40	96.8	95.46667	96.11	10	10	12.63333
	60	96.80111	95.7	96.18667	13	10	11.83333

Table 4-6 Results of the proposed algorithm for Swap = 20

Iterations	Swarm Size	Accuracy			Selected Features		
		Best	Worst	Mean	Best	Worst	Mean
50	20	96.03333	91.96667	94.23	8	12	13.93333
	30	95.09999	92.76667	94.71889	10	17	12.83333
	40	96.3	92.73333	94.83	15	18	13.4

	60	96.46667	93.43333	95.21889	10	15	12.9
100	20	96.13333	93.03333	95.06	14	14	13.13333
	30	97.21111	93.9	95.25333	10	16	12.63333
	40	96.4	93.86667	95.16889	9	13	13.33333
	60	96.56667	94.73333	95.7	12	15	12.83333
250	20	96.33333	94.43333	95.60667	11	10	12.63333
	30	96.4	94.7	95.75333	12	16	12.8
	40	96.5	95.1	95.87111	9	11	12.4
	60	96.83333	95.33333	96.06667	11	12	12.26667
500	20	96.4	95	95.71222	9	12	12.06667
	30	96.7	95.23333	96.01667	14	13	11.96667
	40	96.53333	95.06667	96.03889	10	15	11.5
	60	97.011	95.83333	96.11889	11	13	11.86667

It is clear that the GA-FA algorithm attained good results in terms of classification accuracy. The effect of the swarm size was very clear on the searching process, big swarm size leads to find better results. However, both 40 and 60 number of sources have almost the same accuracies. On another hand, the number of iterations also has the same effect, but the difference between these experiments was very small.

4.4 RESULTS COMPARISON

The proposed IDS model is anomaly-based and has two main stages - the pre-processing stage, which involved the wrapper feature selection process that combines **GA-FA** with the detection classifier (NBC); the second stage is the detection step which showed the performance measures obtained by the classifier with previously selected feature subsets. To test our model, a personal computer with a Core i7 processor, speed 2.2 GHz, and 4 GB of RAM running under Windows 10 operating system was used. Also, for the ranking, the proposed algorithm was benchmarked with three other algorithms (Binary Particle Swarm Optimization (BPSO), Binary Firefly algorithm(BFA), Binary Bat algorithm (BBA)). Grey Wolf Optimizer (GWO), Cuckoo Search Algorithm (CS), hybrid Grey Wolf and Cuckoo Search Algorithm (GWO-CS). The results of these other two algorithms were lifted from previous studies.

All algorithms in this experiment have been examined based on the same swarm size which is equal to 10, and the maximum number of iteration which is equal to 200. Table 4-15 displays the obtained results, with the original accuracy obtained by using NBC.

Table 4-7 The results of all the algorithms

Algorithm	Acc. Rate	Err. Rate	No. Features	Reference
NBC	89.5%	10.5%	41 (ALL)	-
BBA-NBC	91.62%	8.38%	15	(Enache, Sgarciu and Petrescu-Nita, 2015)
BPSO-NBC	90.63%	9.37%	22	(Enache, Sgarciu and Petrescu-Nita, 2015)
GWO	83.54%	16.46%	11	(Xu, Liu, and Su, 2017)
CS	83.54%	16.46%	11	
GWO-CS	83.%	16.46%	6	
BFA-NBC	92.02%	7.98%	14	(Najeeb and Dhannoon, 2018)
GA-FA	97.011	2.09	11	The Proposed Algorithm

It is obvious that the proposed GA-FA algorithm in both objective functions has outperformed the other algorithms in terms of classification accuracy and the number of selected features.

CHAPTER FIVE

CONCLUSION

5.1 INTRODUCTION

This chapter outlines the conclusion of the implementation and findings of the research carried out. This is followed by the future work that can be potentially taken up to broaden the research area of this thesis.

5.2 Conclusion

The development of a machine learning-based IDS requires the design of suitable features that can differentiate normal network activities from intrusion. Despite the number of features previously suggested, the objective evaluation of the earlier proposed schemes is difficult due to the lack of publicly accessible data sets. In a bid to address this issue, the KDDCUP 1999 dataset was formulated by MIT Lincoln laboratory which was further modified by other researchers into the NSL_KDD dataset. Since the formulation of these datasets, they have been the basis for the evaluation of the performance of IDS.

However, there are 41 features in the connection vectors of the raw TCP dump data in the KDD'99 & NSL_KDD datasets; hence, these datasets are considered not suitable for real-time IDS deployment because they contain too many features. This problem has been recently addressed by IDS researches by employing only the parts of the features that have received

much research consideration. This is known as feature selection problems and has driven the desire to develop several feature selection techniques. Feature selection minimizes the computation time and improves the prediction performance; It also helps the understanding of pattern recognition applications or machine learning data.

This study proposed a hybrid wrapper feature selection approach for IDS. The proposed method was evaluated for performance using the NB classifier on the NSL-KDD data set. The evaluation empirically demonstrated the enhancement of the movement & randomization of the FA by the crossover operator imported from GA. The crossover operator helps the best firefly to move to another position by swapping random variables between the best firefly and the second-best firefly. This improvement can translate into better performance in terms of classification performance and the number of features kept in the dataset.

The conclusion of this study could be summarized as follows:

- 1- The NBC achieved a low accuracy (89.5%) when all features were used as inputs.
- 2- The crossover operator enhances the global search ability of Firefly algorithm, in other words, the drawback of the stuck of the best firefly in the swarm was solved using the crossover operator with the second best firefly algorithm.
- 3- The classification accuracy of IDS has been enhanced using NBC classification model when the proposed hybrid algorithm used for selecting the most important subset of features. In general, the results were range in [93%, 97%].
- 4- The impact of the swarm size or the number of fireflies in the population has a notable effect on the performance of the algorithm. It can be seen from the experiments that the classification accuracy was increased when the swarm sized was increased.
- 5- The impact of the number of iterations on the searching process was notable. However, it was not that much as compared to the swarm size.
- 6- The value of *Swap* variable has a notable effect on the results. As the value get increased, the classification accuracy is enhanced as well. In other words, the value of *Swap* bits between the

best firefly and the second best firefly is increased which enhances the chances of the swapping the bits (zeros or ones) between the best two solutions, that led to better exploring the search space. The best found result in this experiment was 97%.

7- The comparison showed the proposed algorithm to outperform the other benchmarking feature selection algorithms.

5.3 FUTURE WORKS

Several applications can be done based on the proposed methodology. The proposed algorithm also can be modified for handling different machine learning tasks and feature selection case studies, as follows:

- Implementing other classifiers as fitness functions rather than NBC, such as SVM, KNN, and ANNs.
- Design and implement of online system to work on the server instead of the offline system. Any new incoming packet can be automatically classified.
- The proposed algorithm can be applied to different feature selection. Such as disease diagnosis, face recognition, and text classification.

REFERENCES

- Ahmed, M., Naser Mahmood, A. and Hu, J. (2016) ‘A survey of network anomaly detection techniques’, *Journal of Network and Computer Applications*, pp. 19–31. doi: 10.1016/j.jnca.2015.11.016.
- Aljawarneh, S., Aldwairi, M. and Yassein, M. B. (2017) ‘Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model’, *Journal of Computational Science*. Elsevier.
- Altwaijry, H. and Algarny, S. (2013) ‘Bayesian based intrusion detection system’, *Lecture Notes in Electrical Engineering*, 170 LNEE(1), pp. 29–44. doi: 10.1007/978-94-007-4786-9_3.
- Amiri, F. *et al.* (2011) ‘Mutual information-based feature selection for intrusion detection systems’, *Journal of Network and Computer Applications*, 34(4), pp. 1184–1199. doi: 10.1016/j.jnca.2011.01.002.
- Amor, N., Benferhat, S. and Elouedi, Z. (2004) ‘Naive bayes vs decision trees in intrusion detection systems’, *ACM Symposium on Applied Computing*, pp. 420–424. doi: 10.1145/967900.967989.
- Arguello, B. (2015) ‘A Survey of Feature Selection Methods : Algorithms and Software A Survey of Feature Selection Methods : Algorithms and Software by’, p. 7.
- Bharti, K. K., Shukla, S. and Jain, S. (2010) ‘Intrusion detection using clustering’, *Proceeding of the Association of Counseling Center Training Agencies (ACCTA)*, 1.
- Bolzoni, D. and Etalle, S. (2008) ‘Approaches in Anomaly-based Network Intrusion Detection Systems’, in *Intrusion Detection Systems*, pp. 1–15. doi: 10.1007/978-0-387-77265-3_1.
- Casas, P., Mazel, J. and Owezarski, P. (2011) ‘UNADA: Unsupervised network anomaly detection using sub-space outliers ranking’, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 40–51. doi: 10.1007/978-3-642-20757-0_4.
- Casas, P., Mazel, J. and Owezarski, P. (2012) ‘Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge’, in *Computer Communications*, pp. 772–783. doi: 10.1016/j.comcom.2012.01.016.
- Chandola, V., Banerjee, A. and Kumar, V. (2009) ‘Anomaly detection: A survey’, *ACM Computing Surveys (CSUR)*, 41(September), pp. 1–58. doi: 10.1145/1541880.1541882.
- Chandola, V., Banerjee, A. and Kumar, V. (2012) ‘Anomaly detection for discrete sequences: A survey’, *IEEE Transactions on Knowledge and Data Engineering*, pp. 823–839. doi: 10.1109/TKDE.2010.235.
- Chandrashekar, G. and Sahin, F. (2014) ‘A survey on feature selection methods’, *Computers and Electrical Engineering*, pp. 16–28. doi: 10.1016/j.compeleceng.2013.11.024.
- Chen, C.-M., Chen, Y.-L. and Lin, H.-C. (2010) ‘An efficient network intrusion detection’, *Computer Communications*, 33(4), pp. 477–484. doi: 10.1016/j.comcom.2009.10.010.
- Chen, Y. *et al.* (2019) ‘Bayesian Personalized Feature Interaction Selection for Factorization Machines’, in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR’19*. New York, New York, USA: ACM Press, pp. 665–674. doi: 10.1145/3331184.3331196.
- Davis, J. J. and Clark, A. J. (2011) ‘Data preprocessing for anomaly based network intrusion detection: A review’, *Computers and Security*, pp. 353–375. doi: 10.1016/j.cose.2011.05.008.

Denning, D. E. (2012) 'An intrusion-detection model', in *Proceedings - IEEE Symposium on Security and Privacy*, pp. 118–131. doi: 10.1109/SP.1986.10010.

Enache, A. C., Sgarciu, V. and Petrescu-Nita, A. (2015) 'Intelligent feature selection method rooted in Binary Bat Algorithm for intrusion detection', in *SACI 2015 - 10th Jubilee IEEE International Symposium on Applied Computational Intelligence and Informatics, Proceedings*, pp. 517–521. doi: 10.1109/SACI.2015.7208259.

Ghorbani, M. A. *et al.* (2017) 'Implementation of a hybrid MLP-FFA model for water level prediction of Lake Egirdir, Turkey', *Stochastic Environmental Research and Risk Assessment*, pp. 1–15. doi: 10.1007/s00477-017-1474-0.

Guan, X., Wang, W. and Zhang, X. (2009) 'Fast intrusion detection based on a non-negative matrix factorization model', *Journal of Network and Computer Applications*, 32(1), pp. 31–44. doi: 10.1016/j.jnca.2008.04.006.

Guyon, I. and Elisseeff, A. (2003) 'An Introduction to Variable and Feature Selection', *Journal of Machine Learning Research (JMLR)*, 3(3), pp. 1157–1182. doi: 10.1016/j.aca.2011.07.027.

Hansman, S. and Hunt, R. (2005) 'A taxonomy of network and computer attacks', *Computers and Security*, 24(1), pp. 31–43. doi: 10.1016/j.cose.2004.06.011.

Hassanzadeh, A. and Sadeghian, B. (2008) 'Intrusion Detection with Data Correlation Relation Graph', *ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*, pp. 982–989. doi: <http://dx.doi.org/10.1109/ARES.2008.119>.

Jovi, A., Brki, K. and Bogunovi, N. (2015) 'A review of feature selection methods with applications', pp. 25–29.

Kabir, M. R., Onik, A. R. and Samad, T. (2017) 'A Network Intrusion Detection Framework based on Bayesian Network using Wrapper Approach', *International Journal of Computer Applications. Foundation of Computer Science*, 166(4).

Kang, S.-H. and Kim, K. J. (2016) 'A feature selection approach to find optimal feature subsets for the network intrusion detection system', *Cluster Computing*, 19(1), pp. 325–333. doi: 10.1007/s10586-015-0527-8.

'KDD Cup 1999 Dataset' (no date). Available at: <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.

Konstantinos, D. and Lazaros, I. (2017) 'A Hybrid Network Anomaly and Intrusion Detection Approach Based on Evolving Spiking Neural Network Classification', in *International Conference on Engineering Applications of Neural Networks*. Cham: Springer, pp. 122–134.

Lee, J. and Kim, D. W. (2015) 'Mutual Information-based multi-label feature selection using interaction information', *Expert Systems with Applications*, 42(4), pp. 2013–2025. doi: 10.1016/j.eswa.2014.09.063.

Liao, H.-J. *et al.* (2012) 'Intrusion detection system: A comprehensive review', *Journal of Network and Computer Applications*, 36, pp. 16–24. doi: 10.1016/j.jnca.2012.09.004.

Liao, H.-J. *et al.* (2013) 'Intrusion detection system: A comprehensive review', *Journal of Network and Computer Applications*, 36(1), pp. 16–24. doi: 10.1016/j.jnca.2012.09.004.

Liao, Y. and Vemuri, V. R. (2002) 'Use of k-nearest neighbor classifier for intrusion detection', *Computers and Security*, 21(5), pp. 439–448. doi: 10.1016/S0167-4048(02)00514-X.

Lin, S. W. *et al.* (2012) 'An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection', *Applied Soft Computing Journal*, 12(10), pp. 3285–3290. doi: 10.1016/j.asoc.2012.05.004.

- Mafarja, M. *et al.* (2017) 'S-Shaped vs. V-Shaped Transfer Functions for Ant Lion Optimization Algorithm in Feature Selection Problem', in *Proceedings of the International Conference on Future Networks and Distributed Systems - ICFNDS '17*. New York, New York, USA: ACM Press, pp. 1–7. doi: 10.1145/3102304.3102325.
- Mafarja, M. and Mirjalili, S. (2018) 'Whale optimization approaches for wrapper feature selection', *Applied Soft Computing*, 62, pp. 441–453. doi: 10.1016/j.asoc.2017.11.006.
- Mitchell, R. and Chen, I. (2014) 'A Survey of Intrusion Detection Techniques for Cyber-Physical Systems', *ACM Computing Surveys (CSUR)*, 46(4), p. 55:1-29. doi: 10.1145/2542049.
- Mukkamala, S., Janoski, G. and Sung, a. (2002) 'Intrusion detection using neural networks and support vector machines', *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN '02*, pp. 1702–1707. doi: 10.1109/IJCNN.2002.1007774.
- Najeeb, R. F. and Dhannoon, B. N. (2018) 'A Feature Selection Approach Using Binary Firefly algorithm for Network Intrusion Detection System', *ARPN Journal of Engineering and Applied Sciences*, 13(6), pp. 2347–2352.
- Sabhnani, M. and Serpen, G. (2004) 'Why machine learning algorithms fail in misuse detection on KDD intrusion detection data set', *Intelligent Data Analysis*, 8(4), pp. 403–415. doi: 10.1007/978-3-540-88623-5_41.
- Taher, K. A., Mohammed Yasin Jisan, B. and Rahman, M. M. (2019) 'Network intrusion detection using supervised machine learning technique with feature selection', in *1st International Conference on Robotics, Electrical and Signal Processing Techniques, ICREST 2019*. doi: 10.1109/ICREST.2019.8644161.
- Tang, J., Alelyani, S. and Liu, H. (2014) 'Feature Selection for Classification: A Review', *Data Classification: Algorithms and Applications*, pp. 37–64. doi: 10.1.1.409.5195.
- Tavallaee, M. *et al.* (2009) 'A detailed analysis of the KDD CUP 99 data set', in *IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009*. doi: 10.1109/CISDA.2009.5356528.
- Tsai, C. F. *et al.* (2009) 'Intrusion detection by machine learning: A review', *Expert Systems with Applications*, pp. 11994–12000. doi: 10.1016/j.eswa.2009.05.029.
- Vigna, G. and Kemmerer, R. A. (1998) 'Network-Based Intrusion Detection Approach', *Acsac*, pp. 25–34. doi: 10.1109/CSAC.1998.738566.
- Wang, G. *et al.* (2010) 'A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering', *Expert systems with applications*. Elsevier, 37(9), pp. 6225–6232.
- Wu, S. X. and Banzhaf, W. (2010) 'The use of computational intelligence in intrusion detection systems: A review', *Applied Soft Computing*, 10(1), pp. 1–35. doi: https://doi.org/10.1016/j.asoc.2009.06.019.
- Xu, H., Liu, X. and Su, J. (2017) 'An improved grey Wolf optimizer algorithm integrated with Cuckoo Search', in *Proceedings of the 2017 IEEE 9th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2017*. doi: 10.1109/IDAACS.2017.8095129.
- Yang, X.-S. (2010) 'Firefly algorithm, stochastic test functions and design optimisation', *International Journal of Bio-Inspired Computation*. Inderscience Publishers, 2(2), pp. 78–84.
- Yang, X. S. (2008) *Firefly algorithm for multimodal optimization, Nature-Inspired Metaheuristic Algorithms*. Luniver Press.
- Yang, X. S. (2014) *Nature-inspired optimization algorithms*. doi: 10.1016/C2013-0-01368-0.

