A Multi-Stage Intrusion Detection Approach for Network Security

Manish Verma

Roll. 213CS2175

under the guidance of

Prof. Sanjay Kumar Jena

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela – 769 008, India

A Multi-Stage Intrusion Detection Approach for Network Security

Dissertation submitted in

June 2015

to the department of

Computer Science and Engineering

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Master of Technology

by

Manish Verma

(Roll. 213CS2175)

under the supervision of

Prof. Sanjay Kumar Jena

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela – 769 008, India

Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, India. `www.nitrkl.ac.in`

June 1, 2015

Declaration

I certify that

- I have complied with all the benchmark and criteria set by NIT Rourkela Ethical code of conduct.
- The work done in this project is carried out by me under the supervision of my mentor.
- This project has not been submitted to any other institute other than NIT Rourkela.
- I have given due credit and references for any figure, data, table which was being used to carry out this project.

Signature of the Student
Place: Rourkela

Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, India. `www.nitrkl.ac.in`

Dr. Sanjay Kumar Jena
Professor

June 01, 2015

Certificate

This is to certify that the work in the project entitled ***A Multi-Stage Intrusion Detection Approach for Network Security*** by ***Manish Verma*** is a record of an original work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of ***Master of Technology*** in ***Computer Science and Engineering***. Neither this project nor any part of it has been submitted for any degree or academic award elsewhere.

Professor
Sanjay Kumar Jena

Acknowledgment

First of all, I would like to express my deep sense of respect and gratitude towards my supervisor Prof. Sanjay Kumar Jena, who has been the guiding force behind this work. I want to thank him for introducing me to the field of Intrusion Detection and giving me the opportunity to work under him. His undivided faith in this topic and ability to bring out the best of analytical and practical skills in people has been invaluable in tough periods. Without his invaluable advice and assistance it would not have been possible for me to complete this thesis. I am greatly indebted to him for his constant encouragement and invaluable advice in every aspect of my academic life. I consider it my good fortune to have got an opportunity to work with such a wonderful person.

I thank our H.O.D. Prof. Santanu Kumar Rath, Prof. Ruchira Naskar and Prof. Bidyut Kumar Patra for their constant support in my thesis work. They have been great sources of inspiration to me and I thank them from the bottom of my heart.

I would also like to thank all faculty members, PhD scholars specifically Santosh Kumar Sahu, my seniors and juniors and all colleagues to provide me their regular suggestions and encouragements during the whole work.

At last but not the least I am in debt to my family to support me regularly during my hard times.

I wish to thank all faculty members and secretarial staff of the CSE Department for their sympathetic cooperation.

Manish Verma

Abstract

Nowadays, the massive increment in applications running on a computer and excessive in network services forces to take convenient security policies into an account. Many methods of intrusion detection proposed to provide security in a computer system and network using data mining methods. These methods comprise of the outlier, unsupervised and supervised methods. As we know, each data mining method is not able to find different types of attacks. So, for removing this vulnerability, we are using Multi-Stage Intrusion Detection Method that containing outlier, unsupervised and supervised detection approaches for improving the performance and detection accuracy by reducing the false alarms for detection of known and unknown attacks. We have used NSL-KDD, KDD Corrected and GureKDD dataset in our experiment.

We have compared our proposed outlier method $GBBK^+$ with GBBK method and our method gives the same result with the less time complexity. The Unsupervised classification algorithm $k-point$ performing the unnecessary comparison of objects iteratively by reducing number of attributes every time up to the threshold that is improved and named as $k-point^+$. Empirically, the proposed scheme compared with existing methods, and the results shows that the proposed method outperform in term of time complexity and detection accuracy.

Keywords: $GBBK^+$, $k-point^+$, Multi-Stage Intrusion Detection System, outlier detection, SVM;

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Nowadays, the massive increment in applications usingona computer and excessive in a network services forces to take convenient security policies into an account. Security in the field of the computer is necessary because of its virtual environment. Attacks or intrusions come with a computer security by compromising at least one of the goal of security that are confidentiality, integrity and availability on data [1]. Intrusion Detection System is the methodology of observing the occasions happening in a computer system or a network. Indications of violations of computer security approaches or standard security policies are analyzed. An IDS is a software that automatically starts the intrusion detection process [2]. Host-based IDS is used to monitors the host and its objective is to detect the malicious activity on that host only by performed local analysis. Network-based IDS operates on network data flow for a segment of the network. It monitors the network to detecting malicious activity. Many of the times, people $don't$ know that there is a distinction between misuse and intrusion detection. Intrusion detection is referred as the malicious activity from the outside whereas misuse is referred as the malicious activity comes from inside only.

1.1 Techniques of IDS

The most popular IDS techniques are pattern-matching based detection, anomaly-based detection and stateful protocol analysis.

1.1.1 Signature-Based Detection

Pattern-matching Based Detection is the method of checking the stored strings or patterns, which represents a known attack against the string that are observed. This method is simple and using string matching. The current packet details entry is matched to a list of stored patterns. It usually not gives false alarms because of predefined rules. But, it is unable to find the new attacks pattern and modified existing patterns. For example, if the existing pattern is ”*connect_proxy*” which is stored in the rule set, but by changing slightly as ”*proxy_connect*” the detection mechanism unable to detect it. Snort is an example of pattern-matching NIDS that can recognize old attacks.

1.1.2 Anomaly-Based Detection

Anomaly Based Detection is the process of examine system activities that created as normal activities against the events that are observed to identify deviations. For example, the normal activity includes web activity usually in day hours. The primary advantage of this method is that it is effective to find new attacks that occur first time. An initial profile that are supposed to be normal is generated. A dynamic profile, on the other hand, regularly gets updated with additional events. Because of the inherent dynamic behavior of networks and systems, static profiles are not suitable as they get outdated soon. Dynamic profiles do not suffer from this deficiency. This method has the problem of false positives. They often treat benign activity may raise an alarm.

1.1.3 Stateful Protocol Analysis

Stateful Protocol Analysis Method is preparing the log activity that supposed to be accepted by each protocol separately and comparing each observed events against them. This analysis is based on vendor developed profiles whereas anomaly-based detection uses host or network dependent profiles.

1.2 Types of Intruder

Intruders are different types in the organization:

(a) **Masqueraders :** Masqueraders are outsiders from the corporation, and they do not have the authority to use the systems. These type of intruders pass through the system security by accounts of the legitimate user.

(b) **Misfeasors :** Misfeasors are insiders who are having authority to use the systems and try to access services that are not granted to them or sometimes it is granted but misuse their privileges.

(c) **Clandestine :** Clandestine users may be insiders or outsiders. These intruders are the individuals who are having a supervisory control to evade security mechanisms such as access control and audit control.

1.3 Classification Methods

Data mining is a process of extracting knowledge or valid data from dataset. There are many difficulties in the database such as redundant data, missing data, not a specific value of attribute and outliers. Taking into account the machine learning techniques, anomaly-based approach can be arranged in two unique classifications unsupervised and supervised.

1.3.1 Unsupervised anomaly detection

Unsupervised anomaly detection is having a good idea of taking the unlabeled data and on the basis of characteristics, the class label is predict for each object. Because of this property, there is no need of labeled training data. This method is effective for finding new attacks without having any previous knowledge about dataset. But it causes to the false identification as the malicious activity is not significantly changes from normal activity to differentiate.

There are two approaches of unsupervised to detect anomalies- outlier based classification and clustering.

Clustering Based Method

Clustering is a technique used to grouping the data objects based on their similar characteristics. The data objects belongs to the cluster having high similarity whereas the data objects that comes under different clusters having the high dissimilarity. The Clustering method works on unlabeled data. After cluster formation, the objects from one cluster possesses the same property, and dissimilar to other objects from other clusters. To distinguish the label of the object of a cluster is equal with the label of the cluster.

Many of the times assumptions comes into consideration that the number of normal objects is outstanding than malicious objects. So, according to this assumption, the larger cluster belongs to the normal class and smaller cluster belongs to malicious class. But, This is not true always. For example, in case of DDOS attack, the larger cluster belong to attack class which violates the assumption which motivates us to use a supervised technique to label the clusters generated by unsupervised techniques.

Outlier Detection Based Method

Outliers are values of a variable that statistical properties are not matched with the other values. They can severely affect the result of predictive analysis. Depending upon the requirement of the application, outliers are of particular interest. Sometimes the presence of outliers adversely affect the conclusion so need to be eliminated. Sometimes these outliers become the center of interest that are containing important information about the abnormal behavior of a system.

There are many data mining techniques minimizing the influence of outliers or eliminating them. Sometimes the consequence are loss of important hidden information since one $person's$ noise could be another $person's$ signal.

The application of Outlier detection includes intrusion detection system, identification of new diseases, financial applications and Credit card fraud detection where outliers may indicate fraudulent activity. By extracting the most relevant features of network traces, packet flow data and packet header information, we can find the outlier or anomaly behavior of an activity.

An intrusion can be detected by finding the data points using the outlier detection, whose features are distinctly different from the rest of the data. Sometimes outliers be individuals or sometimes groups of objects representing the behavior that is outside the range of what is considered normal. According to the clustering algorithm, outliers are objects that are not exist in clusters of the dataset usually called noise.

The challenge in outlier detection in intrusion detection is to handle the large amount of data of mixed-type that is categorical and numerical data. Therefore, the outlier algorithm should be scalable to apply on a large volume of dataset. If we visualize the outlier result of a dataset using a scatter plot, then they are far away from the normal data points. As a result, outlier detection also known as anomaly or deviation analysis.

Outliers detection are the discovery of unexpected behavior and removal or detection of outliers help in avoiding conclusion. Most of the time outlier detection takes first place such as outliers may be generated due to rarely typical events showing entirely different characteristics, deliberate actions, measurement impairment and intentionally actions, etc.

There are various methods in the literature of detecting outliers.

(a) **Distance based :**

In this method, we are counting the number of objects covered in a limited range (threshold) from a point q. If this count is more than a predefined set value of number of objects, then q is considered as normal and otherwise outlier.

(b) **Density-based :**

In this method, we are measure density of a point q by finding the average density of its neighboring objects. Local outliers will find based on the local density of the points, and it depends on its K nearest neighbor points. This score assigned to every point, and it is called Local Outlier Factor (LOF). Points that are having larger LOF will be considered as outliers. But this method is not working properly in sparse dataset.

(c) **Nearest neighbor based :**

In this method firstly finding the distance from each point p to every other point q, perform sorting on distance in increasing order, first k points are termed as K nearest neighbor for that point p. Some of the points are not coming in the nearest neighbor of any point, so these are classified as outliers and other than that are normal.

The part of thesis is on nearest neighbor based detection of outliers. It is showing that by using this method we can find outliers in dense as well as sparse dataset that cannot be possible by using density based outlier detection.Supervised anomaly detection is the machine learning task that created normal behavior model from purely normal training data. The training data comprises of a set of training examples. All these examples are consisting pairs of input record and related desired output value of that record. So this training dataset is used for classifying new data or new connections. By given the object, if it will not produce the desired output then the system generates an alarm. In a real world, $it's$ not easy to get purely normal dataset. $it's$ a very time-consuming process to acquire each object to classify manually and label them as normal or malicious.

1.3.2 Supervised anomaly detection

Supervised anomaly detection is the machine learning task which takes training data to learn the model. The training data comprises of a set of training examples. All these examples are consisting pairs of input record and related desired output value of that record. So, this training dataset is used for classifying new data with unknown label. Practically, $it's$ not easy to get purely normal dataset. This is a very time-consuming process to acquire each object to classifying manually and label them as normal or malicious.

In real time situations, the attacks can be known or unknown yet. The effective approach for unknown attacks is anomaly-based approach rather than signature-based approach. The critical problem in supervised anomaly-based detection is gathering the purely labeled training dataset. This issue is improved in unsupervised anomaly-based detection and finding novel attacks without any knowledge of labeled training data. For better performance, the unsupervised and supervised anomaly detection approach in a sequence is used to detect each type of intrusions.

1.4 Motivation

The motivation is to develop an efficient model that gives high performance for known, and unknown attacks. The detection effectiveness and accuracy of a single detection approach is normally not good comparatively to the combination of more than one approach which motivated us to implement a multi-stage detection method.

1.5 Contribution

Contribution in our thesis is given below:

(a) An enhanced unsupervised Outlier detection algorithm $GBBK^{+}$ based on GBBK algorithm in [3].

(b) An enhanced unsupervised classification algorithm $k - point^{+}$ based on the $k - point$ algorithm in [3].

(c) A hybrid multi-stage classifier is having better performance in all classes categorization based on outlier detection, unsupervised and supervised algorithm.

1.6 Thesis Organization

The overall thesis is organized into five chapters including the introduction.

Chapter 2 presents the Related work about outlier-detection, unsupervised, supervised and multi-stage detection approach.

Chapter 3 presents the Our multi-stage approach overview comprising problem formulation given in Section 3.1 and Architecture given in section 3.2.

Chapter 4 deals with the outlier-detection approach.

Chapter 5 deals with the unsupervised classifier.

Chapter 6 deals with the supervised classifier.

Chapter 7 Result for NSL-KDD, KDD Corrected and GureKDD dataset.

Chapter 8 presents the concluding remarks with scope for future research work.

Chapter 2

Related Work

The main categorization of classes in any standard intrusion dataset is divided as in Table 2.1. Out of these classes DOS, R2L (Remote to local), U2R (User to root) and Probe are the main attack categories and normal class containing the non-malicious connections [4].

- Literature review on Outlier Detection Method
- Literature review on Unsupervised Based Detection
- Literature review on Supervised Based Detection
- Literature review on Multi Stage Based Detection

2.1 Literature review on Outlier Detection Method

Recently, a few studies have been conducted on outlier detection for small as well as large datasets.

Distance based outlier detection method initialized by the Knorr and Ang [5]. According to them an object q is being an outlier if at most p objects are covered in

Table 2.1: Categories of class distribution

Category	**Definition**
Normal	This is the data in dataset that not having any malicious activity.
DoS	Denial of service attacks or distributed denial of service (DDoS) attacks are the attempts done by an attacker to make services unavailable for valid users. E.g. smurf, land and pod.
R2L	Intruder gains unauthorized access from a remote machine to the victim machine. These type of intruders are known as masqueraders. E.g. phf, imap and spy.
U2R	Intruder is already having an authority to use the system and tries use services that are not granted to that user. These type of intruder are known as misfeasor. E.g. rootkit and loadmodule.
Probe	An attacker is attempting access to a computer system and its files from a weak point of the target host. E.g. satan and nmap.

distance d of q. This method gives better performance than depth-based method but highly depend on number of dimensions. Further, Ramaswamy et al. [6], extended the distance-based outliers by using the DK and providing the rank to outliers, where DK (q) represent the distance of the k-nearest neighbor of q. Top-n points with maximum DK considered as outliers, but it is still distance-based.

Some clustering algorithm such as DBSCAN, CURE, and BIRCH can manage outliers as an extent to do not interfere in clustering process [7]. Ranking the priority of outliers by using these algorithms is not possible.

Density Based outlier detection method was initialized by Breunig et al. [8]. According to this method, we are finding the local outlier factor (LOF) value for each data and this value relies on the density of its neighborhood. This LOF value can be used to rank the points regarding their outlier ness.

Casas et al. [9] introduced a UNADA (Unsupervised network anomaly detection algorithm) for malicious traffic. The authors proposed a cluster-based method based on a subspace density clustering technique that detect the outlier in multiple low dimension space. The result came from all multiple clustering is combined to produce ranking outliers of traffic flow.

Aggarwal and Yu [10] introduced a novel techniques for outlier detection that is suitable for low as well as high dimensional datasets. In this method first finding locally sparse lower dimensional projections. As it is locally sparse so by brute force method, cannot be searched possibly because number of combinations are so large. In the distance-based outlier detection, dimensionality is a curse, and this technique removes this difficulty. This implementation almost works as a brute force method in related to finding projections with very negative coefficients, but the cost is lower than the previous methods.

Williams et al. [11], proposed RNN (replicator neural networks) for outlier detection. This method for outlier detection is compared with other methods on data mining datasets that are typically larger and real datasets. It is satisfactory to small as well as large datasets. RNN performed much better on the KDD intrusion datasets.

There are some other methods like Nearest Neighbor factor [12] that relies on nearest neighbor scheme. In this method by finding the average KNN distance for all points in the datasets, it will measure nearest neighbor factor for all points by taking into consideration previously estimated KNN and the average KNN distance. NDoT by the voting mechanism measured outlier ness of each point in the dataset and finally compared with the threshold. The points that are having factor value more than the threshold comes into the outlier list.

Prasanta et al. [3] proposed GBBK method to find outliers on the basis of k-nearest neighbor. To find K nearest neighbors, the author used the quick sort that increases the time complexity of this method. Clusters created a group of similar values and having intra similarity within cluster and dissimilarity between different clusters. UNADA (Unsupervised Network Anomaly Detection algorithm) using sub-space outliers ranking published by Cases et al. [5]. By using sub-space density, novel clustering have been used by it to the purpose of finding different outliers and different clusters in multiple low-dimensionality space.

Other clustering algorithm like BIRCH, CURE and DBSCAN can find outliers without the help of clustering process [13] but outliers ranking cannot be done by these algorithms.

2.2 Literature review on Unsupervised Based Detection

Many methods have been proposed for unsupervised anomaly-based detection. A method for NIDS presented by Kingsly et al. [14]. The grid-based method used in this algorithm based on subspace clustering. All clustering performed on grid structure, and this structure is divided the space into many finite small cells.

An extended BIRCH [15] have been proposed by Burbeck and Nadjm in [16]. ADWICE (Anomaly detection with Real-Time Incremental Clustering) is developed for increasing scalability, fast and adaptive anomaly detection.

2.3 Literature review on Supervised Based Detection

Several times a study have been conducted on supervised anomaly detection. Audit Data Analysis and Mining(ADAM) based [17] on online NIDS. ADAM uses a data mining methods such as association rules. It uses previously trained labeled data to classify upcoming connections as an already known attack, and novel attacks. It contains two phase i.e training and online phase. In the training phase, the model is trained by attack-free data, and it gives the output as rules based profiles of normal activity. After that, this profile is given to another model as an input and training data whose containing malicious data is also inserted in it, and the result comes out the all suspicious activity. So, the malicious data is labeled as attacks and the labeled data is given as an input to a classifier to train it. In the online phase, the test data is given to the system, and this data is detected malicious activity based on previously created rule-based profiles. The detected malicious activity is categorized as a known and unknown attack, and false alarm. ADAM consisted three modules: the first module is preprocessing engine and this module extracting information from the header of TCP/IP packet of each connection established. The second module is engine mining that applied two-phase association rules that discussed above to connections, and the third module is classification engine that classify test data as malicious and non-malicious.

2.4 Literature review on Multi Stage Based Detection

A number of multi-stage IDS have been proposed by different authors to remove intrusions from dataset as much as possible by combining different approaches of data mining, neural network, machine learning, etc. The hybrid IDS introduced by Aydyn et al. in [18] generated by combining two approaches: packet header anomaly detection and network traffic anomaly detection. The paper proposed by Zhang et al. [19] using random forest algorithm comprises rule-based detection and anomaly-based detection. The hybrid intrusion detection proposed by Hwang et al. [20] combines the features of the low false positive rate of misuse based IDS and the anomaly-based detection for the unknown type of attacks. Ozgur Depren et al. [21] proposed an intelligent IDS that having architecture by utilizing the features of anomaly detection method and signature detection method with a decision support system that combining the outcome of both the modules. The hybrid intrusion detection model proposed by Pan et al. [22] combined neural network techniques for classification ability and C4.5 algorithm to find attacks. Multiple-level tree classifier used as IDS rely on different properties of particular intrusion category proposed by Xiang et al. [23].

Table 2.2 describes comparison of existing IDS with criteria such as detection nature (real time or non-real time), detection type (Network based or Host based) and attacks handled (KDDCUP99, DARPA98 etc.) and analysis approach.

Table 2.2: Comparison of IDS

Method Name	Author Name	Publication Year	Detection Nature	Detection Types	IDS Technique	Datasets used/handled	Methodology
Outlier Based	Duan et al. [24]	2008	Non-real time	Network Based	Anomaly Based	KDDCUP99	Clustering
	Otey et al. [25]	2005	Non-real time	Network Based	Anomaly Based	KDDCUP99	Distance Based
	Petrovskiy et al. [26]	2003	Non-real time	Network Based	Anomaly Based	KDDCUP99	Kernel Function
	Breunig et al. [8]	2000	Non-real time	Network Based	Anomaly Based	NHL96	Density Based
Unsupervised	Kuang et al. [27]	2007	Real time	Network Based	Anomaly Based	KDDCUP99	K-Nearest Neighbor
	Burbeck et al. [16]	2005	Real time	Network Based	Anomaly Based	KDDCUP99	Clustering
	Mohajerani et al. [28]	2003	Non-Real time	Network Based	Anomaly Based	KDDCUP99	Neuro fuzzy logic
	Subramoniam et al. [29]	2003	Real Time	Hybrid	Rule Based and Anomaly Based both	KDDCUP99	Statistical
	Sequeira et al. [30]	2002	Real time	Host Based	Anomaly Based	csh history file mechanism	Clustering
	Zhang et al. [31]	2001	Real time	Network Based	Anomaly Based	KDDCUP99	Statistical and Neural nets
Supervised	Horng et al. [32]	2011	Non-real time	Network Based	Anomaly Based	KDDCUP99	Support Vector Machine and Hierarchical Clustering
	Song et al. [33]	2006	Real time	Network Based	Anomaly Based	KDDCUP99	Statistical
	Ertoz et al. [34]	2003	Real time	Network Based	Anomaly Based	KDDCUP99	Classification
	Labib et al. [35]	2002	Real time	Network Based	Anomaly Based	KDDCUP99	Neural nets
	Daniel et al. [17]	2001	Real time	Network Based	Anomaly Based	DARPA98	Association Rule

	Dickerson et al. [36]	2000	Non-real time	Network Based	Anomaly Based	KDDCUP99	Fuzzy logic
	Roesch et al. [37]	1999	Real time	Network Based	Rule Based	KDDCUP99	Rule Based
Multi-level	P. Gogoi et al. [3]	2013	Non-real time	Network Based	Rule Based and Anomaly Based both	KDDCUP99	Decision Tree
	Hui Lu et al. [38]	2009	Non-real time	Network Based	Anomaly Based	KDDCUP99	Decision Tree
	Hwang et al. [20]	2007	Non-real time	Network Based	Rule Based and Anomaly Based both	KDDCUP99	Rule Based
	Zhang et al. [19]	2006	Non-real time	Network Based	Rule Based and Anomaly Based both	KDDCUP99	Random forest
	Depren et al. [21]	2005	Non-real time	Network Based	Rule Based and Anomaly Based both	KDDCUP99	Decision Tree and neural nets
	Xiang et al. [23]	2004	Non-real time	Network Based	Rule Based and Anomaly Based both	KDDCUP99	Decision Tree

Chapter 3

Multi Stage Approach Overview

In this chapter, first establishes the problem statement and then architecture shows the solution of the problem.

3.1 Problem Establishment

The detection accuracy and performance of a single stage classifier is not uniformly good for each class distribution. A single detection approach may not provide better accuracy and reduce the cost of the model, so, the appropriate combination of the multiple classifiers for each type of class distribution in a dataset possibly gives the high accurate result.
The problem formation:

Classifiers are given as $M_1, M_2, \ldots, M_K$ and dataset is D with N classes of distribution. The Multi-Stage IDS objective is to arrange these classifiers M_i effectively to give best classification performance for all classes.

3.2 Architecture

In Fig. 3.1, our Multi-Stage IDS that gives high performance and better accurate result to network attacks. The implementation of Multi-Stage IDS containing three stages for detection of attack: an outlier based detection method, an unsupervised detection method and supervised detection method.

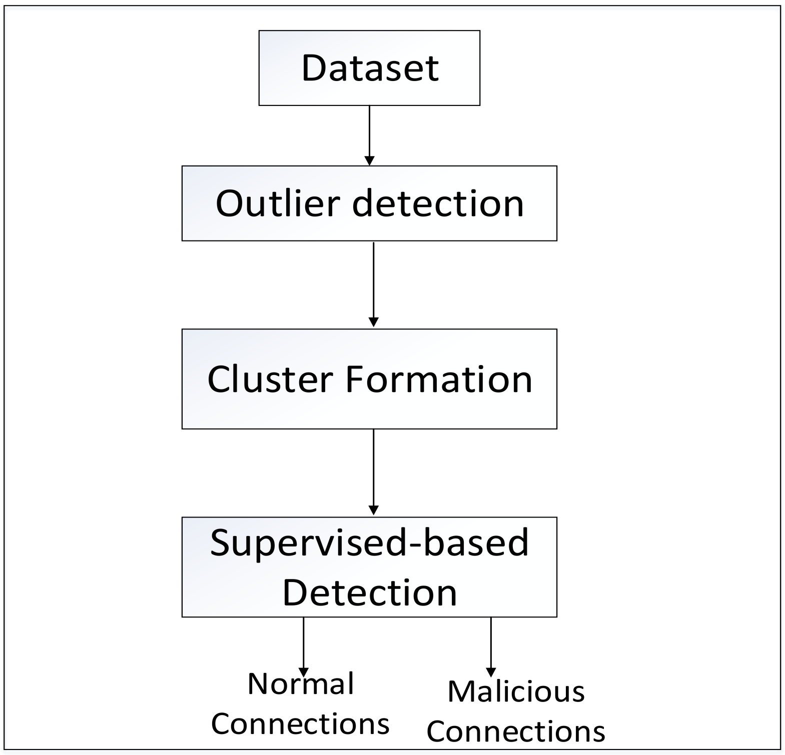

Figure 3.1: Architecture of IDS

The sequence of the classifier at a particular stage depend on the dataset and accuracy of each classifier for different attack categories. In Fig. 3.1, the first stage of architecture categorizes the dataset into two groups i.e. normal and outlier. Outlier detection classifier removes the outliers from the dataset and rest unclassified dataset is given as input to the second stage. This outlier detection is described in Chapter 4 thoroughly. The objective of this stage is to remove outliers and make the dataset more consistent. The second stage is unsupervised based detection classifier, and

$k - point^+$ algorithm is used to form clusters. This stage made the set of clusters based on characteristics of the data distribution. The output of this stage is set of k cluster with an uncategorized cluster which contains the data that not belongs to the k cluster. This second stage is described in Chapter 5 in detail. The third stage is supervised classifier, and it classified the dataset finally into normal or attack class. It takes one random object from each cluster and if the random object is classified as attack then the whole cluster is labeled as an attack otherwise labeled as normal. But we have to take each and every object from unclustered data to label these object separately. This third stage is described in Chapter 6 in brief manner.

The main task is to reduce high dimension set of data to set of the clusters. The supervised stage is capable of classifying the cluster label by taking one instance from the cluster. No need to classify each objects of the cluster. As a result, the detection process is faster and capable to classify the instances more accurately.

Chapter 4

Outlier Detection

The input for this stage is dataset of network connections shown in Fig. 4.1. On the dataset, first we have performed data preprocessing like missing values using [39]. Outliers are separated from the dataset so that output of this stage contains a separate set of outliers that will remove from the dataset and set of connections that are free from outliers will use in next stage for further processing.

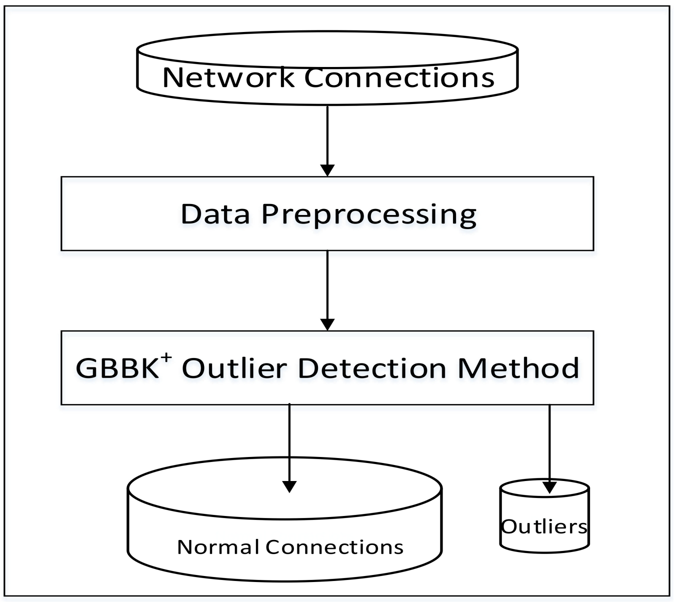

Figure 4.1: Outlier Detection Process

We have developed an outlier detection algorithm that depends on k-nearest neighborhood value [3]. We have calculated Reverse Neighbor Outlier Factor (RNOFk) by finding the nearest neighbor (NNk) and reverse nearest neighbor (RNNk) set of points to identify outliers.

There are different distance measures are discussed in [40] to find the distance between two objects. In our experiment, we have used Euclidean distance to find the distance between two arbitrary points. In the dataset $D = \{P_1, P_2, \ldots, P_n\}$ of n points where $P_1, P_2, \ldots, P_n$ are the objects of the dataset D. Let to find the distance between Pi and Pj, which is represented by d (P_i, P_j).

The Nearest Neighbor Set of k points for a point p is the set of k-nearest neighbor points of p and is denoted by NNk (p) where:

(i) $k > 0$

(ii) p will not come in NNk (p)

(iii) $d(q, p) < d(q', p)$ where q and q' are k^{th} and $(k+1)^{th}$ nearest neighbors of p, respectively.

The Reverse Nearest Neighbor Set of k points for a point p is the set of points that containing p in their NNk and is denoted by RNNk (p) where:

$$RNNK(p) = \{q \,\epsilon\, D \mid p \,\epsilon\, NNk(q)\ ,\ p\ \neq q\} \tag{4.1}$$

We have taken RNNK into account in Equation 4.1, because by using this term we are getting the strength of the link between each point to every other point.

So finally we are calculating the Reverse Neighbor Outlier Factor of that object by taking their Reverse Nearest Neighbor Value. RNOFk is the ratio of number of remaining points by removing the points of RNNk to the number of dataset points.

$$RNOFK(P) = \frac{|D| - |RNNK\_List.RNNK(P)| - 1}{|D| - 1} \tag{4.2}$$

In Equation 4.2, we are subtracting one from numerator and denominator both to exclude that object for which point calculation is going on.

$|D|$ is denoted the number of points in dataset.

In our algorithm, we are using 3 Lists:

(a) **NNk_List:** This is the first phase of our algorithm and every object must goes through this step. After finding the nearest neighbor of k points for every point one by one, place these value into NNk_List. The number of points in NNk (p) always k and number of points are same that is k in every NNk value. This List is the matrix of $|D| \times k$.

(b) **RNNk_List:** This is the second phase of our algorithm and every object must goes through this step also. For finding the RNNk value for each point, we have to scan the whole NNk_List. After making a set of RNNk place these values of all points in RNNk_List. The number of points in RNNk (p) ranging from 0 to $|D|$-1 and number of points are different in every RNNk value. This List is the matrix of $|D| \times (\, range\, from\, 0\, to\, |D| \; - 1\,)$.

(c) **RNOFK_List:** This is the final phase of our algorithm, and it depends on the RNNk_List. In this phase, we are taking a threshold value that depends on the number of objects or points in the dataset, and we will compare this threshold value with RNOFk value of each point. All RNOFk value are placed into RNOFk_List, and it is the matrix of $|D| \times 1$.

4.1 Methodology

In Figure 4.2, a dataset is given by 6 points {A, B, C, D, E, F} and neighborhood size k=3.

First Phase:

NNk (A) = {C, D, E},

NNk (B) = {A, C, F},

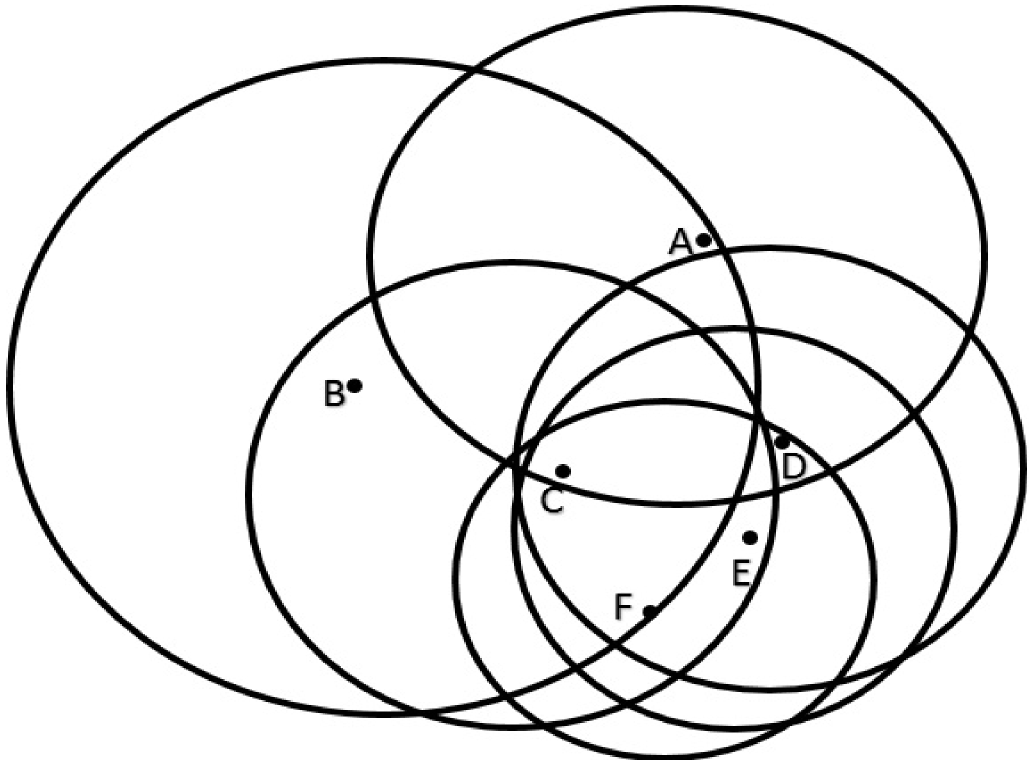

Figure 4.2: Example of Dataset Points

NNk (C) = {B, E, F},

NNk (D) = {C, D, E},

NNk (E) = {C, E, F},

NNk (F) = {C, D, E},

Second Phase:

RNNk (A) = {B},

RNNk (B) = {C},

RNNk (C) = {A, B, D, E, F},

RNNk (D) = {A, E, F},

RNNk (E) = {A, C, D, F},

RNNk (F) = {B, C, D, E},

The number of objects of NNk and RNNk for k=3 as follows:

$\lvert NNk(A)\rvert = 3,$	$\lvert RNNk(A)\rvert = 1,$
$\lvert NNk(B)\rvert = 3,$	$\lvert RNNk(B)\rvert = 1,$
$\lvert NNk(C)\rvert = 3,$	$\lvert RNNk(C)\rvert = 5,$
$\lvert NNk(D)\rvert = 3,$	$\lvert RNNk(D)\rvert = 3,$
$\lvert NNk(E)\rvert = 3,$	$\lvert RNNk(E)\rvert = 4,$
$\lvert NNk(F)\rvert = 3,$	$\lvert RNNk(F)\rvert = 4,$

Third Phase:

$RNOFK(A) = (\frac{6-1-1}{6-1}) = 0.80$

$RNOFK(B) = (\frac{6-1-1}{6-1}) = 0.80$

$RNOFK(C) = (\frac{6-5-1}{6-1}) = 0.0$

$RNOFK(D) = (\frac{6-3-1}{6-1}) = 0.40$

$RNOFK(E) = (\frac{6-4-1}{6-1}) = 0.20$

$RNOFK(F) = (\frac{6-4-1}{6-1}) = 0.20$

Threshold always has a range from 10 to 20 percent of the dataset so accordingly for this dataset threshold (T) = 0.80 Then Outliers are A and B.

4.2 Proposed Algorithm

Our Algorithm contains three functions named as FindNNk (D, k) , FindRNNk (D, NNk_List) and FindRNOFk (D, RNNk_List).

The function FindNNk (D, k) making the distance matrix for all points using Euclidean distance then perform extraction of k shortest distances.

The function FindRNNk (D, NNk_List) searches in the output of the first phase that which points are connected bidirectionally in a neighborhood way.

The function FindRNOFk (D, RNNk_List) computes the outlier ness of each point and on the basis of them we are finding outliers in the dataset.

Algorithm 1 $GBBK^{+}$ Algorithm

```
Input:
    Dataset D,
    Threshold T,
    Nearest Neighbor k,
    Object p.

Process:
  function FindNNk($D, K$)
    while $|D| \neq 0$ do
      $\forall_{q\ where\ p\ ,\ q\ \epsilon\ D\ AND\ p \neq q}$ ,
      Calculate distance from p to q and stores into Dist.
    end while
    for $i \leftarrow 1\ to\ k$ do
      $\forall_{q\ where\ q\ \epsilon\ D}$ , Extract_Min_Update(i) on Dist(p) and placed in NNk(p).
    end for
    $\forall_{p}$ , $NNk(p)\ into\ NNk\_List.$
  end function
  function FindRNNk($D, NNk\_List$)
    while $|D| \neq NULL$ do
      $\forall_{q\ where\ p \neq\ q\ AND\ p\ \epsilon\ NNk\_List.NNk(q)\ AND\ q\ \epsilon\ D}$ ,
      Add q to RNNK (P) that insert into RNNk_List.
    end while
  end function
  function FindRNOFk($D, RNNk\_List$)
    Compute $\forall_{p\ \epsilon\ D}$ , RNOFK (P)= $\frac{|D|-|RNNK\_List.RNNK(P)|-1}{|D|-1}$
    And add to the list RNOFk_List.
  end function
  $\forall_{S\ \epsilon\ RNOFk\_List}$ ,
  if $S \geq T$ then
    Add S to O;
  end if

Output: Outlier List O
```

4.3 Simulation

We have simulated the algorithm using Matlab. Missing values and redundancy records are handled using [39]. The proposed algorithm used to find the outlier present in the dataset. As per the class label, the dataset is divided into number of subsets. The Final output of the algorithm can be obtained by combining the indexes of outliers calculated individually.

4.4 Complexity Analysis

The proposed algorithm consists with three functions namely FindNNk(), FindRNNk() and FindRNOFk(). We can find the time complexity of FindNNk() function by considering the distance among n objects and extract k-shortest distance. As a result, the O(n+n) is the complexity of this function. The time complexity of FindRNNK() function is obtained by searching among $n \times k$ objects. For n object its complexity is $O(n \times k \times n)$. The Function FindRNOFk() had complexity $O(n)$. Thus the time complexity of the proposed algorithm is $O(n+n+(n \times k \times n)+n) = O(n^2)$. However our algorithm and previous algorithm complexity are asymptotically equal but for a very large value of n our algorithm will give the better performance.

4.5 Performance Analysis

As per the literature, the outlier distribution lies in between 10 to 20 percentage of the dataset. The number of outliers relies on the threshold value T. In this experiment, and we have set the value of T as 0.99. The number of outliers inversely proportional to the threshold value T as given in Equation 4.3.

$$No.\ Of\ Outlier\ \alpha\ \frac{1}{T} \tag{4.3}$$

To evaluate the performance of proposed outlier detection algorithm, we have implemented the SVM-based predictive model by applied the dataset before and after removal of the outlier. We train and test the model using these two version of datasets. The Table 4.1 describes the confusion matrix of the model before removal of the outlier, and the Table 4.2 is the confusion matrix of the model after removal of the outlier.

Table 4.1: Confusion matrix of original Dataset before removal of outlier using SVM.

11734	9	99.16%
99	13350	99.92%
99.92%	99.27%	99.5713%

The model classified 25084 instances correctly out of 25192 instances in NSLKDD Train20 dataset. After removing the outliers, the number of instances reduced to 24634 out of which 24530 instances are accurately classified.

Table 4.2: Confusion matrix of original Dataset after removal of outlier using SVM.

11475	9	99.17%
95	13055	99.92%
99.91%	99.27%	99.5778%

The True Positive (1^{st} cell), False Negative (2^{nd}), Precision (3^{rd}), False Positive (4^{th}), True Negative (5^{th}), Negative Predicted Value (6^{th}), Sensitivity (7^{th}), specificity (8^{th}), and Accuracy (9^{th}), are illustrated in the Table 4.1 and 4.2. The accuracy of the model without outlier is more in compare to the accuracy of the model with the outlier. In other words, the training error of the model without outlier is less. Therefore, the model is more efficient on the data without outlier in compare to the data with the outlier. To avoid model biasing and over-fitting the dataset should be outlier free.

To measure the computation time between GBBK and proposed method

$GBBK^{+}$, we have executed 7500 instances with 500 intervals. The execution of number of instances presented in X-axis as given Figure 4.3. The Y-axis contains the estimated time in seconds to complete the execution of the algorithms. This graph in Figure 4.3 shows that as the number of instances increases, the time taken by GBBK is also increasing rapidly compared to proposed method. So for the larger number of instances, $GBBK^{+}$ takes less time in compared to GBBK algorithm.

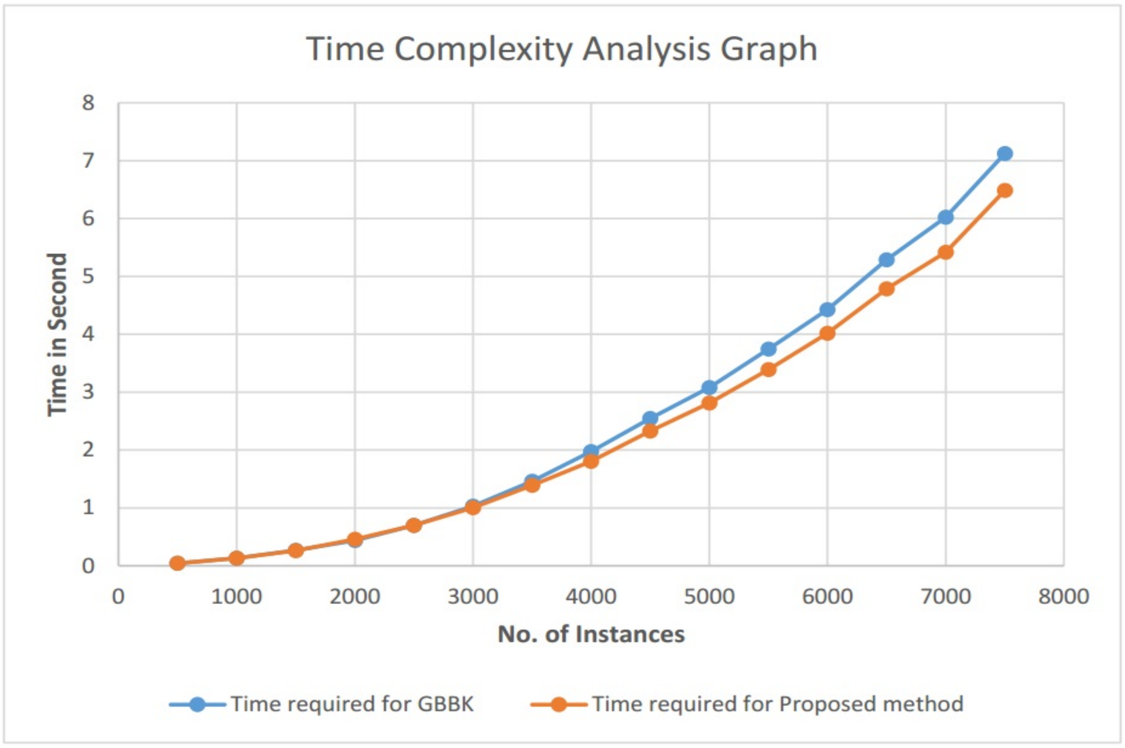

Figure 4.3: Comparison of execution time between GBBK and proposed method $GBBK^{+}$

The Receiver Operating Characteristic (ROC) describes the performance of the classification system. The Figure 4.4, illustrates the ROC of NSLKDD dataset on SVM. In X-axis, the False Positive Rate (FPR) and Y-axis True Positive Rate (TPR) is given. ROC curve is a plot of TPR against FPR. The Figure 4.5, is the ROC of NSLKDD Train dataset without the outlier. Its performance is slightly more in compare to the Figure 4.4.

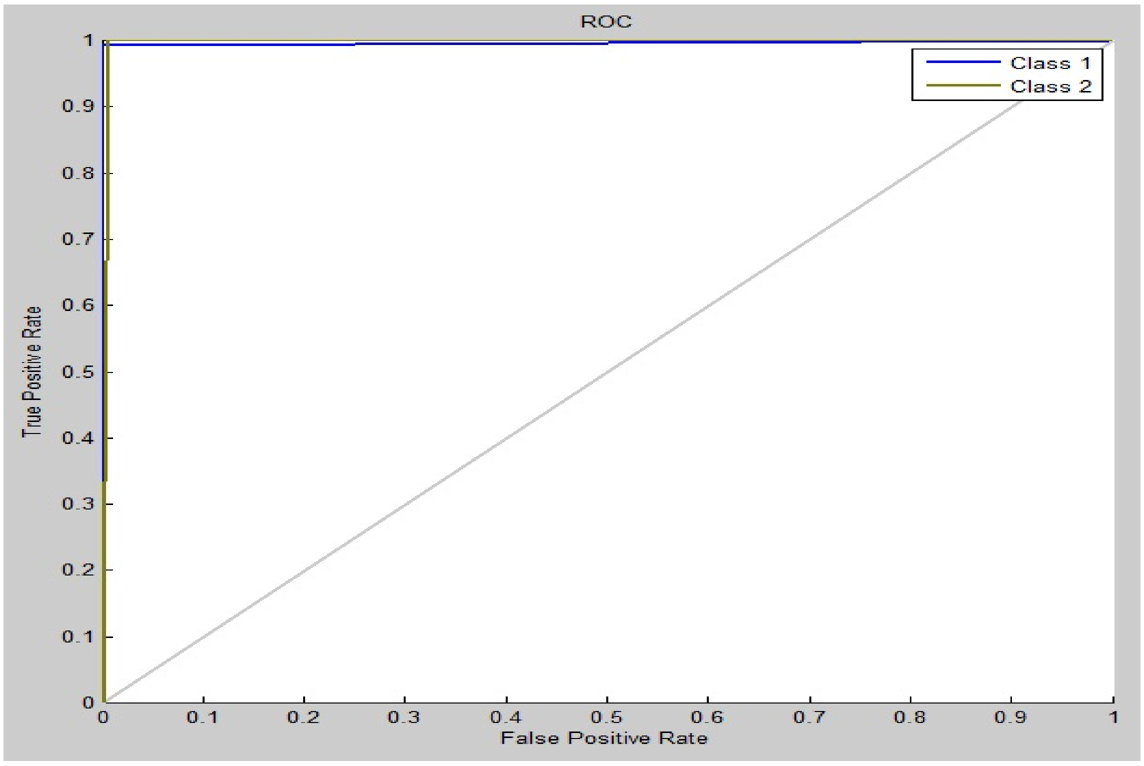

Figure 4.4: ROC before removal of outlier

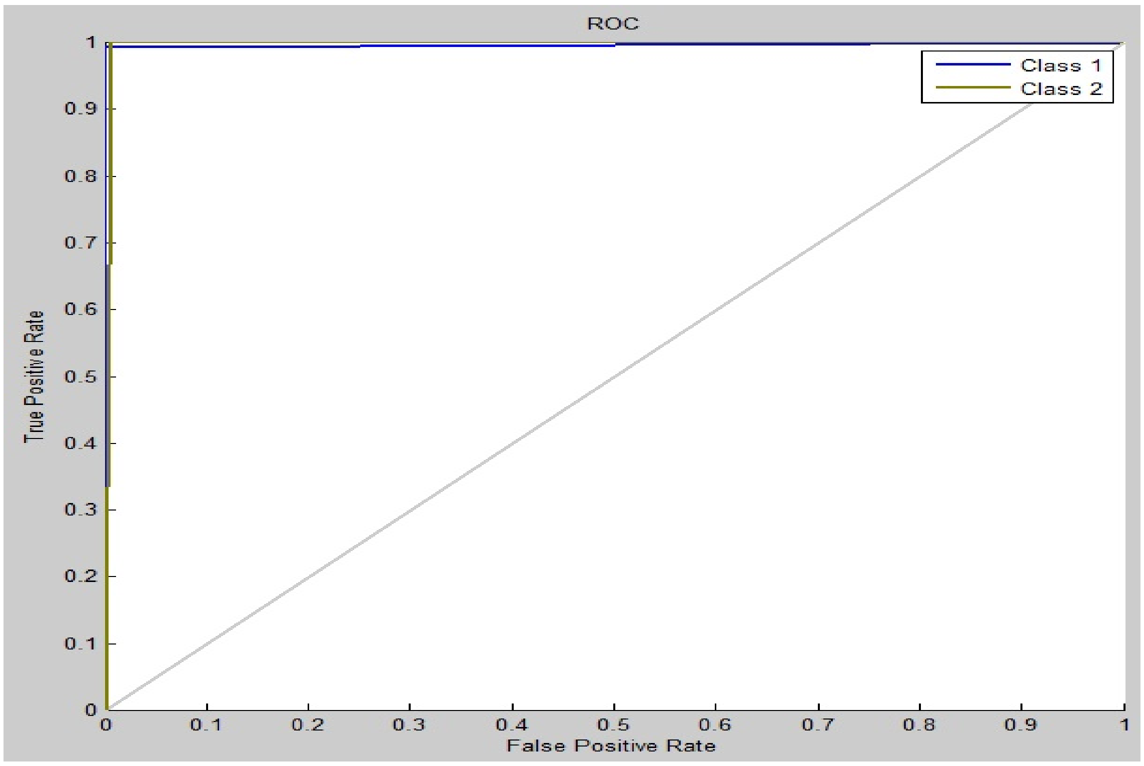

Figure 4.5: ROC after removal of outlier

4.6 Summary

Our proposed algorithm based on K-Nearest Neighbor Outlier detection algorithm that works on neighborhood property by taking distance from both the points to know the interest of each other. We have found the outlier score of an object by measuring Reverse Neighbor Outlier Factor. Our proposed algorithm and its experimental result shows that in consideration of time, it performs better than previous methods. Real-Time data most of the time so large and time is very crucial nowadays. As a result, the objective of the proposed algorithm is to find outliers from a large dataset with least computational time.

Chapter 5

Unsupervised Based Detection

The input for this stage is outlier free dataset of network connections shown in Fig. 5.1. The objective of this stage is creating a set of cluster based on the similarity measure. The output of this stage is (k+1) clusters in which $(k+1)^{th}$ cluster contains the uncategorized objects and in others k clusters, each cluster having same properties.

The unsupervised classification algorithm $k - point$ presented in [3] performing the unnecessary comparison of objects iteratively by reducing number of attributes every time up to the threshold (minimum attribute). One more limitation is that is the largest cluster labeled as normal data but it is not always true. For example, in DOS attack, the larger cluster belong to the malicious class rather than normal. So, these two limitations have been rectified in proposed algorithm and named as $k - point^{+}$.

The proposed algorithm is taken unlabeled data and create a list of clusters, as per inherent statistical property in the data. The k random objects have been chosen from the dataset and all the objects are clustered around these k random objects based on similarity function find_sim() that creates finally (k+1) clusters from the whole unlabeled dataset.

$$find\_sim(x, y) = distance(x,\ y) \tag{5.1}$$

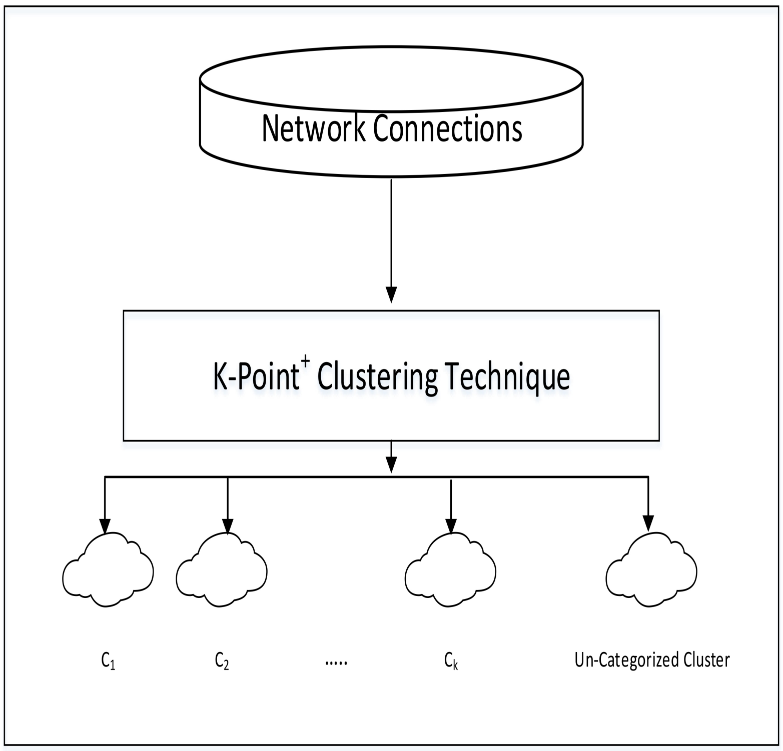

Figure 5.1: Unsupervised Classification Process

The dataset contains N number of objects $\{O_1, O_2, \ldots, O_N\}$ and each object have M number of attributes $\{A_1, A_2, \ldots, A_M\}$. Each attribute A_i has finite domain $\{S_1, S_2, \ldots, S_M\}$. The object O_i is defined as $\{V_1, V_2, \ldots, V_M\}$ where V_1 is the value in domain S_1 of attribute A_1, and V_2 is the value in domain S_2 of attribute A_2 and so on. The idea is more clear by example in Table 5.1.

5.1 Methodology

In Table 5.1, we have taken a dataset that contains 15 objects $\{O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8, O_9, O_{10}, O_{11}, O_{12}, O_{13}, O_{14}, O_{15}\}$ with 10 attributes named as $\{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, A_9, A_{10}\}$ and domain for attributes are given in Table 5.2 as follows:

Table 5.1: Example of Dataset

Object No. / Attribute No.	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
O_1	1	1	3	5	1	2	5	3	1	3
O_2	2	2	1	4	2	3	4	5	1	4
O_3	3	2	3	1	3	1	2	6	2	5
O_4	2	1	2	2	1	2	3	4	1	7
O_5	3	2	2	2	3	3	1	2	2	2
O_6	2	1	1	1	1	1	3	1	2	1
O_7	3	2	2	3	1	3	1	6	1	2
O_8	1	1	1	1	3	1	2	6	2	3
O_9	3	2	2	3	2	1	2	2	1	6
O_{10}	2	2	1	2	2	3	1	1	4	4
O_{11}	3	2	2	4	3	2	1	3	4	1
O_{12}	3	1	1	5	1	2	1	3	2	3
O_{13}	2	1	1	4	1	3	2	4	3	5
O_{14}	1	1	3	5	3	1	1	5	4	7
O_{15}	2	2	3	5	2	1	2	6	3	7

we have selected the value of $k = 4$. Hence, 4 clusters $\{C_1, C_2, C_3, C_4\}$ are formed with satisfying the condition $min\_att = \frac{M}{3}$. In table 5.3, all 4 clusters with the object numbers is formed. Here, Rank shows the stability of the object in that cluster and No._Of_Attribute shows the similar attributes of the cluster and object. But here one more cluster has been made that is C_0 that contains those Uncategorized objects that not comes in any cluster $\{C_1, C_2, C_3, C_4\}$ and formed in single cluster C_0.

Table 5.2: Domain Table

Domain Name	**Values of Domain**
S_1	1, 2, 3
S_2	1, 2
S_3	1, 2, 3
S_4	1, 2, 3, 4, 5
S_5	1, 2, 3
S_6	1, 2, 3
S_7	1, 2, 3, 4, 5
S_8	1, 2, 3, 4, 5, 6
S_9	1, 2, 3, 4
S_{10}	1, 2, 3, 4, 5, 6, 7

5.2 Proposed $k - point^+$ Algorithm

Initially, the algorithm not contains any cluster. Select k unique objects randomly from the dataset where k is the number of clusters to be form. All the objects O_i read from the dataset sequentially and compare with existing k clusters. If it is matched with any cluster based on similarity measure without violating the predefined threshold value of min_att (minimum number of attribute must match to insert in that cluster which may vary dataset to dataset) then assign the object to that cluster.

The objects that not satisfied the similarity condition as per the threshold min_att with any cluster, assign those in the uncategorized cluster. The output of this algorithm is a list of clusters and every cluster grouped all similar objects with the similarity measure. Some of the objects are uncategorized as per the threshold value. Total k+1 clusters are formed, where k is predefined and one additional cluster due to the instances that are not satisfied the given similarity condition.

find_sim ( ) function given in Equation 5.1, find similarity by comparing the

Algorithm 2 $k - point^{+}$ Algorithm

Input:

```
    Dataset D that containing N objects with M attributes,
    $D' = Unique\ records\ in\ dataset\ D$ ,
    $K = No.\ of\ instances\ randomly\ selected\ from\ D'$ ,
    $min\_att = \frac{M}{3}(depend\ upon\ the\ values\ in\ Dataset)$.
```

Process:

```
  while $|D| \neq NULL$ do
      Select an object O from D.
      for $r \leftarrow 1$ to $k$ do
          sim_mat = find_sim (O, r)
      end for
      $\forall_i$ , select such $k_i$ which has maximum similarity in sim_mat, and store the
  score V along with its cluster index I.
      if $V \geq min\_att$ then
          Rank [O] = $\frac{V}{M}$
          Cluster_index = I
      else
          Rank [O] = $\frac{V}{M}$
          Cluster_index = 0 (Uncategorized Cluster)
      end if
  end while
```

Output: A list of (k+1) clusters in which $(k+1)^{th}$ cluster contains uncategorized objects.

Table 5.3: Clusters List

Cluster_Index	**Object_List**	**Rank**	**No._Of_Attribute**
C_1	O_2	0.6	6
	O_{10}	1	10
C_2	O_4	0.4	4
	O_6	1	10
	O_8	0.5	5
	O_{13}	0.4	4
C_3	O_1	0.6	6
	O_{12}	1	10
C_4	O_3	0.4	4
	O_5	1	10
	O_7	0.6	6
	O_9	0.4	4
	O_{11}	0.5	5
C_0	O_{14}	0.3	3
	O_{15}	0.3	3

values of the attributes present in r and O. it returns 0 if the values are equal, otherwise returns a non-zero value. For numeric attributes, it simply subtracts one value to other, and for categorical attributes it subtracts as per their ASCII values. The find_sim ( ) returns the similarity score in the form of zero and non-zero values and stores it into sim_mat. Select an instance among k records and returns its similarity score V and index I.

V is compared with the threshold min_att. If satisfied then the rank ($\frac{V}{M}$) and the cluster index I is assigned to that object. The rank gives the stability of the object in that cluster. Otherwise, the objects belong to the Uncategorized cluster with index 0. Finally, the algorithm returns k+1 clusters.

5.3 Simulation

The algorithm 2 is implemented in Matlab 2015a with the system configured as Intel i7 CPU with 3.4 GHZ, 14GB RAM and Windows 8.1 64 bit Operating System. Three datasets are used as input which are described in Chapter 7. Data preprocessing is done as per [39]. The algorithm 2 takes dataset D, k as number random points as input and generate k+1 number of clusters as output. The first k clusters formed as per the rank of the similar measure, and the $(k+1)^{th}$ formed due to the instances are not satisfied the threshold min_att.

5.4 Complexity Analysis

The $k - point^{+}$ algorithm scan the dataset once only. Every object is compared with k number of the cluster for M number of attributes. So, the time complexity is O (NkM) where N is the number of objects in dataset. In the previous algorithm $k - point$, the comparison is done by decreasing the number of attributes iteratively for similarity measure but here we are finding the difference between both the objects and count the number of $0's$ comes. If the count of zeroes is greater than the threshold min_att then we can insert in that cluster. By this way, no need to check iteratively between two objects and we can do it with only one iteration. As number of attributes increasing, our algorithm gives better time complexity than previous.

Chapter 6

Supervised Based Detection

In Supervised Classification, Support Vector Machine (SVM) is used for classification model that trained by labeled training dataset. The output of unsupervised classifier is a list of k+1 clusters is given as an input to supervised classifier, out of which first k clusters are labeled by predicting only one random object from whole cluster. If this random object is classified as an attack, then the entire cluster is labeled as attack otherwise labeled as normal as shown in Fig. 6.1. But, $(k+1)^{th}$ cluster contains uncategorized objects $that's$ why examine of each object is needed in this cluster to define the label of each object separately. Hence, number of object input to the model is k + number of object in $(k+1)^{th}$ cluster. The first's k predicted label is assign as the label of k cluster and then assign the cluster label to its individual object's class label. After label assignment to all objects, they are categorize as normal and malicious.The IDS allow the normal connections and drop or alert the malicious connections to the administrator. The support vector machine works in two steps shown in Fig. 6.2: online phase and offline phase.

Offline phase also known as training phase. Three datasets (KDD99, NSLKDD, and GureKDD) we have used in our experiment and all these datasets are first combined and then cross-validation to train the model. The datasets have the feature set, and corresponding class label (purely labeled trained datasets) and by using these labeled datasets SVM-Model is trained.

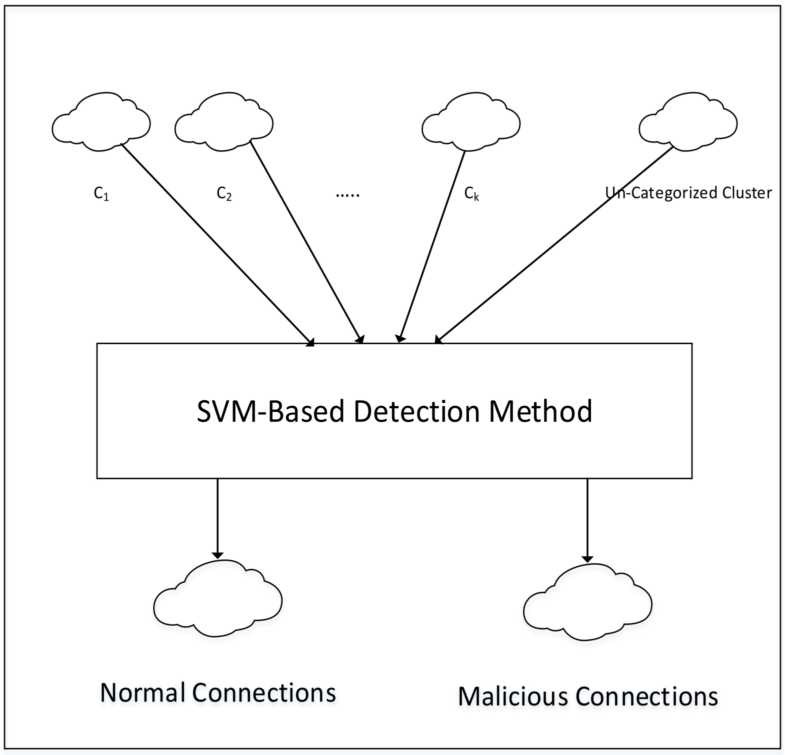

Figure 6.1: Supervised Classification Process

The online phase also known as testing phase. In this phase, the model formed in offline phase is used for prediction. The output of $k - point^+$ algorithm (set of (k+1) clusters) that containing feature set only and SVM-Based prediction is used to predict the class label of these feature sets. We have used binary classifier SVM model that gives the final output in two classes: the normal class that contains normal network connection and allowed them. The other class is the malicious class that contains all malicious network connections and blocked them or submit this report to the administrator.

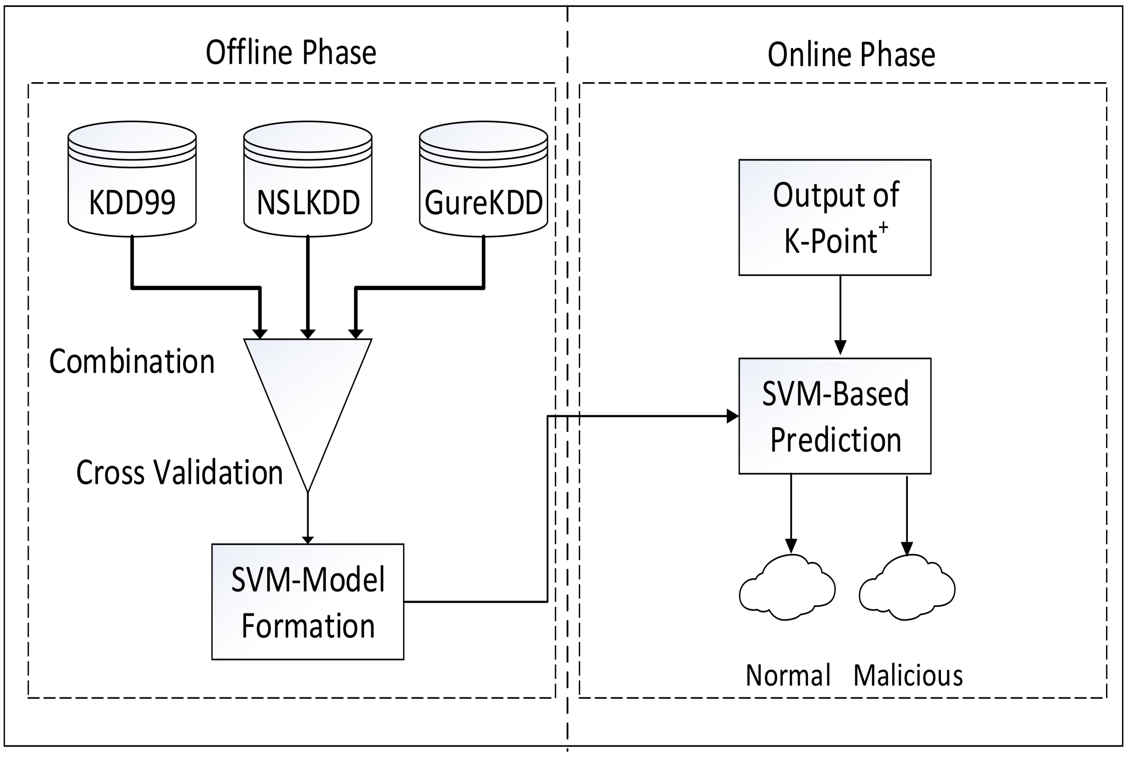

Figure 6.2: Offline and Online phase in SVM Model

Chapter 7

Result and Implementation

We have used KDDCorrected, NSL-KDD_Full and GureKDD dataset. Each dataset have different classes, and each class have different number of records. The Table 7.1 contains the dataset names horizontally and classes names vertically. The Table 7.1 also contains the number of objects present and outliers detected class-wise as per the algorithm 1.

Some of the classes are skipped because that are not contains significant number of values in any dataset and comes in the row of skipped objects. The last row shows the total records of the dataset and out of which total outliers of that dataset. These outlier indexes will be removed from the original dataset to further processing by next stage.

Table 7.1: Outlier report of different datasets

Dataset Name /Class Name	**NSLKDD20**		**NSLKDD_Full**		**KDD_corrected**		**GureKDD**	
	No. of records	**No. of Outliers**	**No. of records**	**No. of Outliers**	**No. of records**	**No. of Outliers**	**No. of records**	**No. of outlier**
Back	196	3	956	39	1098	515	0	0
Nmap	301	11	1493	87	84	0	0	0
ipsweep	710	24	3599	64	306	84	0	0
neptune	8282	363	41214	5722	58001	25688	0	0
normal	13449	2544	67343	11176	60593	1025	174873	13847
portsweep	587	28	2931	176	354	133	0	0
satan	691	32	3633	190	1633	564	0	0
smurf	529	17	2646	190	164091	12568	0	0
teardrop	188	3	892	50	12	0	1085	685
warezclient	181	5	890	50	0	0	1749	95
buffer_overflow	6	0	30	0	22	0	0	0
ftp_write	1	0	8	0	3	0	0	0
guess_passwd	10	0	53	0	4367	2729	50	0
imap	5	0	11	0	1	0	0	0
land	1	0	18	0	9	0	0	0
loadmodule	1	0	9	0	2	0	0	0
multihop	2	0	7	0	18	0	0	0
phf	2	0	4	0	2	0	0	0
pod	38	0	201	0	87	0	0	0
rootkit	4	0	10	0	13	0	29	0
spy	1	0	2	0	0	0	0	0
warezmaster	7	0	20	0	1602	475	19	0
perl	0	0	3	0	2	0	0	0
saint	0	0	0	0	736	232	0	0
processtable	0	0	0	0	759	9	0	0
mscan	0	0	0	0	1053	55	0	0
mailbomb	0	0	0	0	5000	399	0	0
dict	0	0	0	0	0	0	879	69
guest	0	0	0	0	0	0	50	2
apache2	0	0	0	0	794	31	0	0
skipped object	0	0	0	0	0	0	174	0
Total	25192	3030	125973	17744	299848	44476	178908	14698

The dataset contains normal and attack network connections. Three dataset listed below are used in out experiment:

- **KDDCup 99:** The KDDCup 99 dataset is used for Third KDD tool Competition held on 1999. It is available in three different versions. The lastest updated version is kddcorrected dataset. In this experiment, kddcorrected dataset is used. The Table **??** describes the number of instances, attributes and available class present in the dataset.

- **GureKDD:** The team GureKDD follow the same steps followed by the team KDDCup 99 dataset. The DARPA98 tcpdump files are processed using Bro IDS and generate the connections and labelled using connection-class files(tcpdump.list) provided by MIT. This is available in two formats i.e. GureKDD Full and GureKDD6 percent.In this experiment we have used GureKDD6 percent dataset. The detail statistical informations is given in Table **??**.

- **NSL-KDD:** The NSL-KDD dataset is a refined version of KDD 10 percent and kddcorrected dataset. The author combined the two datasets and applied data preprocessing techniques. The dataset available in three versions namely NSLKDD Full, NSLKDD 20 percent and NSLKDD Test dataset. In our experiment NSLKDD Full dataset is used. The details informations of this dataset is given in Table **??**.

Table 7.2: Intrusion Datasets

Dataset	No. of objects	No. Features	No. objects without Outlier	No. of class distribution	Release year
KDD Corrected	311029	41	288016	38	1999
NSLKDD	125973	41	108252	23	2005
GKDD	178810	41	164798	28	2007

The $K - point^+$ algorithm formed k+1 number of clusters among which one instance of each total k number of objects are selected from k clusters and all instances of $(k+1)^{th}$ cluster are selected for SVM based supervised model to label the clusters as normal or intrusive class. The confusion matrix for the overall model is given in Table 7.3, Table 7.4, and Table 7.5 for GureKDD, NSL-KDD and KDDCorrected dataset respectively.

As per the Table 7.3, the True Negative and False Negative value are zero. The reason behind this is, the Gurekdd dataset is highly imbalanced. Out of 178810 instances 174000 are normal connections and remaining are distributed among twenty seven class which are misclassified by the model. As a result the accuracy of the predictive model only depend on the normal instances and ignores the malicious connections.

Table 7.3: Confusion Matrix of GureKDD dataset

	Predicted Class	1	-1
Actual Class	1	161026	0
	-1	3772	0

In Table 7.4 the confusion matrix of NSL-KDD dataset is given.

Table 7.4: Confusion Matrix of NSL_KDD_Full dataset

	Predicted Class	1	-1
Actual Class	1	41798	10287
	-1	1842	54325

The confusion matrix of KDDCorrected dataset is given in Table 7.5.

Table 7.5: Confusion Matrix of KDD_Corrected dataset

	Predicted Class	1	-1
Actual Class	1	237258	5088
	-1	44181	1489

The overall outcome of the proposed detection approach is given in Table 7.6. The Performance of a classifier are evaluated as per parameters given in Table 7.6.

To achieve high classification accuracy, the data distributions should be balanced. Otherwise, the class which is highly distribution provides major contribution during model formation and the least class distributions are ignoring by the model during prediction.

Table 7.6: Accuracy of Datasets

Dataset	**Accuracy**	**Recall**	**Specificity**	**NPV**	**FPR**	**Precision**	**MCC**	**F_Measure**
GureKDD	97.711	1	0	0	1	0.9771	0	0.9884
NSL_KDD	88.795	0.9672	0.9672	0.84	0.0328	0.9578	0.7840	0.8733
KDD_Corrected	82.8937	0.9790	0.0326	0.2264	0.9674	0.8430	0.0284	0.9059

Chapter 8

Conclusions and Future Work

The proposed model shows a multi stage IDS rely on outlier , unsupervised and supervised detection approach. The $GBBK^+$ algorithm provides better result in term of time and detection accuracy in comparison to GBBK algorithm. The outlier detection approach detect and delete the outliers from the dataset.As a result the next level performs better because the dataset is free from outliers by the first approach.The outlier detection approach can effectively handle high dimension data with least computational time.

Our proposed algorithm $k-point^+$ rectified the limitations and disadvantages of previous $k-point$ algorithm. The empirical result shows that $k-point^+$ outperform in term of time complexity and detection accuracy than $k-point$. The proposed model is capable to detect less frequency attack as well as high frequency attacks of both known as well as unknown attacks.

In future, we will attempt to make an optimized model that is more effective in terms of time speed and detection capability of attacks that will develop a contribution in the study of intrusion detection.

Bibliography

[1] Rebecca Bace and Peter Mell. Nist special publication on intrusion detection systems. Technical report, DTIC Document, 2001.

[2] Behrouz A. Forouzan. *Introduction to Cryptography and Network Security*. McGraw-Hill Higher Education, 2008.

[3] Prasanta Gogoi, DK Bhattacharyya, Bhogeswar Borah, and Jugal K Kalita. Mlh-ids: a multi-level hybrid intrusion detection method. *The Computer Journal*, 57(4):602–623, 2014.

[4] Richard P Lippmann, David J Fried, Isaac Graf, Joshua W Haines, Kristopher R Kendall, David McClung, Dan Weber, Seth E Webster, Dan Wyschogrod, Robert K Cunningham, et al. Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation. In *DARPA Information Survivability Conference and Exposition, 2000. DISCEX'00. Proceedings*, volume 2, pages 12–26. IEEE, 2000.

[5] Edwin M Knox and Raymond T Ng. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the International Conference on Very Large Data Bases*, pages 392–403. Citeseer, 1998.

[6] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD Record*, volume 29, pages 427–438. ACM, 2000.

[7] Wen Jin, Anthony KH Tung, Jiawei Han, and Wei Wang. Ranking outliers using symmetric neighborhood relationship. In *Advances in Knowledge Discovery and Data Mining*, pages 577–593. Springer, 2006.

[8] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000.

[9] Pedro Casas, Johan Mazel, and Philippe Owezarski. Unada: Unsupervised network anomaly detection using sub-space outliers ranking. In Jordi Domingo-Pascual, Pietro Manzoni, Sergio Palazzo, Ana Pont, and Caterina Scoglio, editors, *NETWORKING 2011*, volume 6640 of *Lecture Notes in Computer Science*, pages 40–51. Springer Berlin Heidelberg, 2011.

[10] Charu Aggarwal and S Yu. An effective and efficient algorithm for high-dimensional outlier detection. *The VLDB JournalThe International Journal on Very Large Data Bases*, 14(2):211–221, 2005.

[11] Graham Williams, Rohan Baxter, Hongxing He, Simon Hawkins, and Lifang Gu. A comparative study of rnn for outlier detection in data mining. In *2013 IEEE 13th International Conference on Data Mining*, pages 709–709. IEEE Computer Society, 2002.

[12] Neminath Hubballi, BidyutKr. Patra, and Sukumar Nandi. Ndot: Nearest neighbor distance based outlier detection technique. In *Pattern Recognition and Machine Intelligence*, volume 6744 of *Lecture Notes in Computer Science*, pages 36–42. Springer Berlin Heidelberg, 2011.

[13] Zuriana Abu Bakar, Rosmayati Mohemad, Akbar Ahmad, and Mustafa Mat Deris. A comparative study for outlier detection techniques in data mining. In *Cybernetics and Intelligent Systems, 2006 IEEE Conference on*, pages 1–6. IEEE, 2006.

[14] Kingsly Leung and Christopher Leckie. Unsupervised anomaly detection in network intrusion detection using clusters. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, pages 333–342. Australian Computer Society, Inc., 2005.

[15] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996.

[16] Kalle Burbeck and Simin Nadjm-Tehrani. Adwice anomaly detection with real-time incremental clustering. In *Information Security and Cryptology ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 407–424. Springer Berlin Heidelberg, 2005.

[17] Daniel Barbará, Julia Couto, Sushil Jajodia, and Ningning Wu. Adam: a testbed for exploring the use of data mining in intrusion detection. *ACM Sigmod Record*, 30(4):15–24, 2001.

[18] M Ali Aydın, A Halim Zaim, and K Gökhan Ceylan. A hybrid intrusion detection system design for computer network security. *Computers & Electrical Engineering*, 35(3):517–526, 2009.

[19] Jiong Zhang and Mohammad Zulkernine. A hybrid network intrusion detection technique using random forests. In *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*, pages 8–pp. IEEE, 2006.

[20] Kai Hwang, Min Cai, Ying Chen, and Min Qin. Hybrid intrusion detection with weighted signature generation over anomalous internet episodes. *Dependable and Secure Computing, IEEE Transactions on*, 4(1):41–55, 2007.

[21] Ozgur Depren, Murat Topallar, Emin Anarim, and M Kemal Ciliz. An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks. *Expert systems with Applications*, 29(4):713–722, 2005.

[22] Zhi-Song Pan, Song-Can Chen, Gen-Bao Hu, and Dao-Qiang Zhang. Hybrid neural network and c4. 5 for misuse detection. In *Machine Learning and Cybernetics, 2003 International Conference on*, volume 4, pages 2463–2467. IEEE, 2003.

[23] C Xiang, MY Chong, and HL Zhu. Design of mnitiple-level tree classifiers for intrusion detection system. In *Cybernetics and Intelligent Systems, 2004 IEEE Conference on*, volume 2, pages 873–878. IEEE, 2004.

[24] Lian Duan, Lida Xu, Ying Liu, and Jun Lee. Cluster-based outlier detection. *Annals of Operations Research*, 168(1):151–168, 2009.

[25] Matthew Eric Otey, Srinivasan Parthasarathy, and Amol Ghoting. Fast lightweight outlier detection in mixed-attribute data. *Techincal Report, OSU–CISRC–6/05–TR43*, 2005.

[26] MI Petrovskiy. Outlier detection algorithms in data mining systems. *Programming and Computer Software*, 29(4):228–237, 2003.

[27] Liwei (vivian Kuang. Dnids: A dependable network intrusion detection system using the csi-knn algorithm, 2007.

[28] M. Mohajerani, Ali Moeini, and M. Kianie. Nfids: a neuro-fuzzy intrusion detection system. In *Electronics, Circuits and Systems, 2003. ICECS 2003. Proceedings of the 2003 10th IEEE International Conference on*, volume 1, pages 348–351 Vol.1, Dec 2003.

[29] N Subramanian, Pramod S Pawar, Mayank Bhatnagar, Nihar S Khedekar, Srinivas Guntupalli, N Satyanarayana, VK Vijaykumar, Praveen K Ampatt, Rajiv Ranjan, and Prasad J Pandit. Development of a comprehensive intrusion detection system–challenges and approaches. In *Information Systems Security*, pages 332–335. Springer, 2005.

[30] Karlton Sequeira and Mohammed Zaki. Admit: anomaly-based data mining for intrusions. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 386–395. ACM, 2002.

[31] Zheng Zhang, Jun Li, CN Manikopoulos, Jay Jorgenson, and Jose Ucles. Hide: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification. In *Proc. IEEE Workshop on Information Assurance and Security*, pages 85–90, 2001.

[32] Shi-Jinn Horng, Ming-Yang Su, Yuan-Hsin Chen, Tzong-Wann Kao, Rong-Jian Chen, Jui-Lin Lai, and Citra Dwi Perkasa. A novel intrusion detection system based on hierarchical clustering and support vector machines. *Expert systems with Applications*, 38(1):306–313, 2011.

[33] Sui Song, Li Ling, and CN Manikopoulo. Flow-based statistical aggregation schemes for network anomaly detection. In *Networking, Sensing and Control, 2006. ICNSC'06. Proceedings of the 2006 IEEE International Conference on*, pages 786–791. IEEE, 2006.

[34] Levent Ertoz, Eric Eilertson, Aleksandar Lazarevic, Pang-Ning Tan, Vipin Kumar, Jaideep Srivastava, and Paul Dokas. Minds-minnesota intrusion detection system. *Next Generation Data Mining*, pages 199–218, 2004.

[35] Khaled Labib and V. Rao Vemuri. Nsom: A tool to detect denial of service attacks using self-organizing maps, 2003.

[36] John E Dickerson and Julie A Dickerson. Fuzzy network profiling for intrusion detection. In *Fuzzy Information Processing Society, 2000. NAFIPS. 19th International Conference of the North American*, pages 301–306. IEEE, 2000.

[37] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *LISA*, volume 99, pages 229–238, 1999.

[38] Hui Lu and Jinhua Xu. Three-level hybrid intrusion detection system. In *Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*, pages 1–4. IEEE, 2009.

[39] Santosh Kumar Sahu, Sauravranjan Sarangi, and Sanjaya Kumar Jena. A detail analysis on intrusion detection datasets. In *Advance Computing Conference (IACC), 2014 IEEE International*, pages 1348–1353. IEEE, 2014.

[40] Santosh Kumar Sahu and Sanjay Kumar Jena. A study of k-means and c-means clustering algorithms for intrusion detection product development. 2014.

Dissemination

Journal

1. Manish Verma, Santosh Kumar Sahu,Sanjay Kumar Jena,"A Multi-Stage Intrusion Detection Approach for Network Security" ,Neurocomputing, Elsevier, May 2015(Communicated).

Conference

1. Manish Verma, Santosh Kumar Sahu,Sanjay Kumar Jena,"K-NN based outlier detection technique on intrusion dataset" ,International Conference on Futuristic Trends in Computational analysis and Knowledge management, Greater Noida, *IEEE 2015*.