International Journal of Engineering Research and Development
e-ISSN: 2278-067X, **p**-ISSN: 2278-800X, www.ijerd.com
Volume 8, Issue 2 (August 2013), PP. 23-38

Feature Extraction Based Classification Technique for Intrusion Detection System

Manish Kumar Nagle[1], Dr. Setu Kumar Chaturvedi[2]
[1] Research Scholar, Technocrats Institute of Technology, RGPV, Bhopal.
[2]Professor & Head Dept. of Comp. Sci. & Engg., Technocrats Institute of Technology, RGPV, Bhopal.

Abstract:- The growth of internet environment has also achieved to increase in end user suspicious activities. Every user gets connected to the network environment which growths unauthorized activities in the system. For protecting data from unauthorized activities or detecting intrusions, there is a necessity to implement security mechanism for identifying unauthorized probable sign of events. Intrusion Detection System (IDS) is used for finding the above activities. Intrusion detection is the process of intelligently monitoring the system activities for identifying the conceivable signs of attack. So the primary aim of Intrusion Detection Systems (IDS) is to protect the availability, confidentiality and integrity of network information systems. In this research classifier has been applied using Naïve Bayes, Bagging, Boosting, Stacking, and J48 on five attack categories as found in the NSL-KDD dataset intrusion detection dataset for novelty attacks as well as for Original dataset and prepossessed dataset. It compares the performance of different classification algorithms which may be categorized into five broad attacks namely Normal, Probe, DoS, U2R, and R2L.

Keywords:- Intrusion Detection System (IDS), Classification Techniques, Information Gain (IG), Network Attack, Evolution Metrics.

I. INTRODUCTION

Today the internet and computer system has become a part of daily life and an essential tool. It utilities people in many areas, such as business, education, medical, entertainment etc. The openness and scalability of the internet has made it a flexible platform for a new generation of on-line services, such as E-commerce, military, social network, public web services, stock prices, online shopping, online reservation etc. The popularity of these services has caused in a huge volume of financial transactions and other type of sensitive information being accessed via the internet. Internet has elevated numerous security issues due to the explosive use of network, the importance and value of this information and the related on-line services which has made the internet a board for a wide variety of attacks and threatens its security of the internet [5, 7].

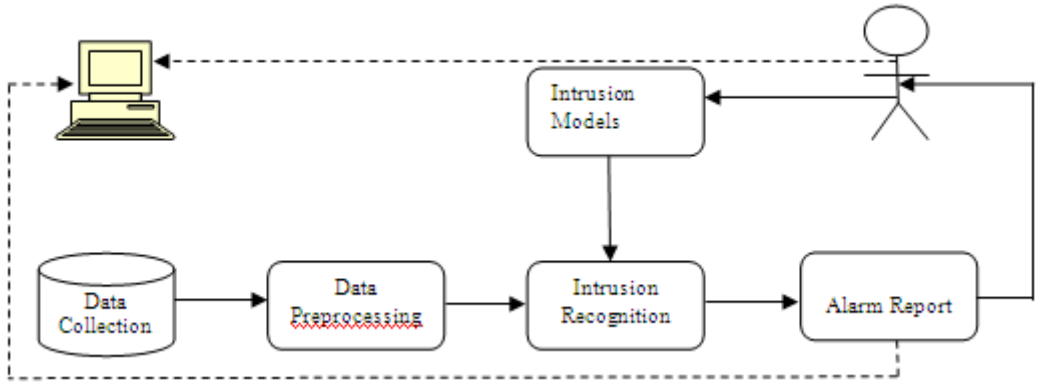

Figure 1 depicts the organization of IDS where solid lines indicate data/control flow while dash lines indicate response to intrusive activities [42].

1.1 Network Security

Network security consists of the provision and policies adopted by a network administrator to prevent and monitor unauthorized access, misuse, modification or denial of a computer network and network accessible resources [43].

1.2 Intrusion and Intruder

Intrusion means breaking in to a computer system or network and then misuses them and performs the malicious activities. When a user of an information system takes an action which the user is not legally permitted to perform is called Intrusion [43].

Intruder is the person who breaks the computer system or network and misuses the computer system or network is called as an Intruder. There are two types of Intruders namely External Intruder and Internal Intruder [43].

External Intrusion comes from outside and cause damages to computer system or network. External intruders do not have any authorized access to the system they attack. Hackers are the example of external intruders [3].Internal Intruder is an insider who exceeds his limited authority to take action. His action may or may not be harmful to the health of the system or the services provided by the system but it seeks to gain added ability to take action without authentic authorization. Internal intruders may act within or outside their limits of authorization [8, 19].

1.3 Intrusion Detection System (IDS)

Intrusion Detection System is the procedure of monitoring and analysing the events occurring in a computer system in order to detect signs of security problems, a security measure that helps to identify a set of malicious actions that compromise the integrity, confidentiality and availability of information resources. Intrusion detection is a difficult problem because of the trade-off thought of detection accuracy, detection speed, the dynamic environment of the networks and the available processing power for processing high volumes of data from distributed networked systems [9].

1.4 Detection Methodologies

Intrusion detection methodologies are classified in following three major categories:

(i) Signature-based Detection (SD)
(ii) Anomaly-based Detection (AD)

1.4.1 Signature-based Detection (Knowledge-based)

A signature-based detection (SD) is a pattern or string that corresponds to a known attack or threat. SD is the process to compare pattern against captured proceedings for recognizing likely intrusions. Because of using the knowledge accumulated by exact attacks and system vulnerabilities, SD is also known as Knowledge-based Detection or Misuse Detection [13, 22].

1.4.2 Anomaly-based Detection (Behavior-based)

An Anomaly-based Detection (AD) is a deviation to a known behavior and profiles signify the normal or expected behaviors derived from monitoring regular activities, network connections, host or users over a period of time. Profile may be either statics or dynamic and usage, the count of e-mails sent etc. AD compares normal profiles with observed event to identify significant attacks. The examples of AD are attempted break-in, masquerading, penetration by legitimate user, Denial-of Service (DoS), Trojan horse, etc [6, 19].

1.5 Type of Intrusion Detection System

There are several types of intrusion detection systems and the choice of which one to use depends on the inclusive risks to the organization and the resources available. One of the classifications of IDS is established by the resource they monitor. According to this classification, basically IDS divided into two categories. There are two types of IDS: Host-based Intrusion Detection System (HIDS) and Network-based Intrusion Detection System (NIDS). A HIDS resides on particular host and looks for indications of attacks on the host. A NIDS resides on a separate system that watches network traffic, looking for indications of attacks that traverse the specified part of the network [14,20].

1.5.1 Host Based Intrusion Detection System

A Host Based Intrusion Detection System (HIDS) monitors the characteristics such as network traffic, system logs, running processes, application activity, file access and modification and system application configuration modification. This is most usually deployed on critical host such as publicly nearby servers and server containing sensitive information [14, 21]. An HIDS exists as a software process on a system. Usually HIDS systems have examined log entries for specific information. More recently, a new form of HIDS has been created that examines calls to the operating system kernel. This type of HIDS is programmed with known attack signature and will give alarm if a system call matches any of the signatures. Both types of HIDS are accomplished of checking file on the system for modification. This is done by execution a cryptographic checksum on the file using a hashing function such as MD5. This value is then stored and used as a comparison against periodic checksum of the file. If the checksum do not match, the file has been possibly altered and the HIDS will report this information [38].

1.5.2 Network Based Intrusion Detection System

A Network Based Intrusion Detection System (NIDS) exists as a software process on a dedicated hardware system into promiscuous mode, which means that the card passes over all traffic on the network to the NIDS software. The traffic is then analyzed according to a set of rules and attack signatures to determine if it is so traffic of interest. If it is an event is generated. The most common configuration for an NIDS is to use two network interface cards. One card is used to monitor a network. This card is placed in a ‘stealthy’ mode so that it does not have an IP address and therefore, does not respond to incoming connections. The stealthy card does not have a protocol stack bound to it so that it cannot respond to analyses such as a ping. The second card is used to

communication with the IDS management system and to send alarms. This card attached to an internet network that is not visible to the network being monitored [42].

1.6 Networking Attacks

The four main categories of networking attack are following see attacks on a network may comfortably be placed into one of these groupings.

Denial of Service (DoS): A DoS attack is a type of attack in which the hacker makes a computing or memory resources too busy or too full to serve legitimate networking requests and hence repudiating users access to a machine e.g. apache, smurf, neptune, ping of death, mail bomb, UDP storm etc. are all DoS attacks [12, 44].

Remote to Local/User Attacks (R2L): Attackers does not have an account on the target machine, hence tries to gain access, these are guess_passwd, ftp_write, multihop, phf, spy, imap, warezclient and warezmaster. [12, 23].

User to Root Attacks (U2R): These attacks are misuses in which the hacker starts off on the system with a normal user account and attempts to abuse the vulnerabilities in the system in order to gain super user privileges e.g. perl, xterm etc [12].

Probing: Probing is an attack in which the hacker scans a machine or a networking device in order to determine weaknesses or vulnerabilities which may later be exploited so as to compromise with the system. This technique is commonly used in data mining e.g. saint, portsweep, mscan, nmap etc.[12]

1.7 Classification Algorithms

Classification is the problem of identifying which of a set of categories (sub-populations) a novel observation belongs, on the basis of a training dataset containing observations (or instances) whose category membership is known. This technique used to expect group membership for data instances [25].

1.7.1 Naïve Bayes Classification

Naïve Bayes is a simple technique for classification using a simple probabilistic model from Bayes theorem with the assumptions of independent attributes. Naïve Bayes is a type of supervised learning algorithm that uses a maximum likelihood method for parameter estimation. It requires a set of training data to estimate means and variances of the attributes for classification.

The Naïve Bayesian Classifier, or simple Bayesian classifier, works as follows:

1. Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, $\boldsymbol{X} = (x_1, x_2, \ldots, x_n)$, depicting n measurements made on the tuple from n attributes, respectively $A_1, A_2, \ldots, A_n$.

2. Suppose that there are m classes, $C_1, C_2, \ldots, C_m$. Given a tuple, $\boldsymbol{X}$, the classifier will predict that $\boldsymbol{X}$ belongs to the class having the highest posterior probability, conditioned on $\boldsymbol{X}$. That is, the naïve Bayesian classifier predicts that tuple $\boldsymbol{X}$ belongs to the class Ci if and only if

$$P(C_i|X) > P(C_j|X) \quad \text{for } 1 \le j \le m, j \ne i.$$

Thus maximize P(C|X). The class C_i for which P(C|X) is maximized is called the maximum posteriori hypothesis.

$$P(C_i|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|C_i)P(C_i)}{P(\boldsymbol{X})}.$$

3. As $P(\boldsymbol{X})$ is constant for all classes, only $P(\boldsymbol{X}|C_i)P(C_i)$ need be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely, that is, $P(C_1) = P(C_2) = \ldots\ldots = P(C_m)$, and would therefore maximize $P(\boldsymbol{X}|C_i)$. Otherwise, maximize $P(\boldsymbol{X}|C_i)P(C_i)$. Note that the class prior probabilities may be estimated by equally likely, that is, $P(C_1) = P(C_2) = \ldots = P(C_m)$, and would therefore maximize $P(\boldsymbol{X}_j|C_i)$. Otherwise, maximize $P(\boldsymbol{X}|C_i)P(C_i)$. Note that the class prior probabilities may be estimated by $P(C_i)=|C_{i,D}|/|D|$,where $|C_{i,D}|$ is the number of training tuples of class C_i in D.

4. Given data sets with many attributes, it would be extremely computationally expensive to compute $P(\boldsymbol{X}|C_i)$. In order to reduce computation in evaluating $P(\boldsymbol{X}|C_i)$, the naïve assumption of class conditional independence is made. This presumes that the values of the attributes are conditionally independent of one another, given the class label of the tuple (i.e., that there are no dependence relationships among the attributes). Thus,

$$\begin{aligned} P(\boldsymbol{X}|C_i) &= \prod_{k=1}^{n} P(x_k|C_i) \\ &= P(x_1|C_i) \times P(x_2|C_i) \times \cdots \times P(x_n|C_i). \end{aligned}$$

The probabilities $P(x_1|C_i), P(x_2|C_i), \ldots, P(x_n|C_i)$ from the training tuples. Recall that here X_k refers to the value of attribute A_k for tuple $\boldsymbol{X}$.

1.7.2 Bagging Classifier

Bagging, which means bootstrap aggregation, is one of the simplest but most successful ensemble methods for improving unstable classification problems. The bagging technique is very useful for large and high-dimensional data, such as intrusion data sets methods for improving unstable classification problems.

Algorithm: Bagging. The bagging algorithm creates an ensemble of models (classifiers or predictors) for a learning scheme where each model gives an equally-weighted prediction.

Input:
D, a set of d training tuples;
k, the number of models in the ensemble;
a learning scheme (e.g., decision tree algorithm, back propagation, etc.)
Output: A composite model, *M*.
Method:

(1) for *i* = 1 to *k* do // create *k* models:
(2) create bootstrap sample, *Di*, by sampling *D* with replacement;
(3) use *Di* to derive a model, *Mi*;
(4) end for

To use the composite model on a tuple, ***X***:

(1) if classification then
(2) let each of the *k* models classify ***X*** and return the majority vote;
(3) if prediction then
(4) let each of the *k* models predict a value for ***X*** and return the average predicted value.

1.7.3 Boosting Classifier

Boosting is an ensemble method for boosting the performance of a set of weak classifiers into a strong classifier. This technique can be viewed as a model averaging method and it was originally designed for classification, but it can also be applied to regression. Boosting provides sequential learning of the predictors.
Algorithm: Adaboost: A boosting algorithm creates an ensemble of classifiers. Each one gives a weighted vote.
Input:
D, a set of d class-labeled training tuples;
k, the number of rounds (one classifier is generated per round);
a classification learning scheme.
Output: A composite model.
Method:

(1) initialize the weight of each tuple in *D* to 1=*d*;
(2) for *i* = 1 to *k* do // for each round:
(3) sample *D* with replacement according to the tuple weights to obtain *Di*;
(4) use training set Di to derive a model, Mi;
(5) compute error(Mi), the error rate of Mi
(6) if error(Mi) > 0:5 then
(7) reinitialize the weights to 1=d
(8) go back to step 3 and try again;
(9) endif
(10) for each tuple in *Di* that was correctly classified do
(11) multiply the weight of the tuple by *error*(*Mi*)=(1-*error*(*Mi*)); // update weights
(12) normalize the weight of each tuple;
(13) endfor

To use the composite model to classify tuple, ***X***:

(1) initialize weight of each class to 0;
(2) for *i* = 1 to *k* do // for each classifier:
(3) *wi* = *log* [1-*error*(*Mi*)/*error*(*Mi*)] ; // weight of the classifier's vote
(4) *c* = *Mi*(***X***); // get class prediction for ***X*** from *Mi*
(5) add *wi* to weight for class *c*
(6) endfor
(7) return the class with the largest weight.

1.7.4 Stacking Classification

Stacking is the abbreviation to refer to Stacked Generalization. Unlike bagging and boosting it uses different learning algorithms to generate the ensemble of classifiers. The main idea of stacking is classifiers from different learners such as decision trees, instance-based learners etc. Since each one uses different knowledge representation and different learning biases the theory space will be explored differently and different classifiers will be found. Thus, it is expected that they will not be correlated.

When the classifiers have been generated they must be combined. Unlike bagging and boosting, stacking does not use a voting system because, for example, if the majority of the classifiers make evil predictions this will lead to a final bad classification. To resolve this problem stacking uses the concept of Meta learner [34]. One way to outputs is by voting the same mechanism used in bagging. However (unweight) voting only makes sense if the learning schemes perform comparably well. If two of the three classifiers make predictions that are completely incorrect, trouble instead stacking introduces the concept of a Meta learner, which replaces the voting procedure. The problem with voting is that it's not clear which classifier to trust.

Stacking tries to learn which classifiers are the reliable ones, using another learning algorithm the Meta learner to discover how best to combine the output of the base learners [25].

The input to the Meta model also called the level-1 model is the predictions of the base models, or level-0 models. A level-1 instance has as many attributes as there are level-0 learners, and the attribute values give the predictions of these learners on the corresponding level-0 instance. When the stacked learner is used for classification, an instance is first fed into the level-0 models, and each one guesses a class value. These guesses are fed into the level-1 model, which combines them into the final prediction.

1.7.5 J48 Decision Trees Classification

A decision tree is a predictive machine-learning model that decides the target value (dependent variable) of a new sample based on various attribute values of the available data. The internal nodes of a decision tree represent the different attributes the branches between the nodes tell the possible values that these attributes can have in the observed samples while the terminal nodes tell the final value (classification) of the dependent variable.

The attribute that is to be predicted is known as the dependent variable since its value depends upon or is decided by the values of all the new attributes. The new attributes which help in predicting the value of the dependent variable are known as the independent variables in the dataset.

The J48 Decision tree classifier follows the following simple algorithm. In order to classify a novel item it first needs to create a decision tree based on the attribute values of the obtainable training data. So, whenever it encounters a set of items (training set) it finds the attribute that discriminates the several instances most clearly. This feature that is able to tell us most nearby the data instances so that classify them the best is said to have the highest information gain [4]. Now, among the possible values of this feature, if there is any value for which there is no ambiguity that is for which the data instances falling within its category have the same value for the target variable then terminate that branch and allocate to it the target value that have obtained [25].

For the other cases, then look for another attribute that gives the highest information gain. Hence continue in this method until either gets a clear decision of what combination of attributes gives a specific target value, or run out of attributes. In the event that run out of attributes, or if cannot get an unambiguous result from the available information, assign this branch a target value that the majority of the items under this branch possess [36].

1.8 Feature Selection Algorithms.

Attribute selection also known as feature selection is the process of selecting a subset of the terms occurring in the training set and using only this subset as features in text classification. Feature selection serves two main purposes [35].

1. It makes training and applying a classifier more efficient by decreasing the size of the effective vocabulary.
2. Feature selection often increases classification accuracy by eliminating noise features (A *noise feature* is one that, when added to the document representation, increases the classification error on new data).

1.8.1 Information Gain Attribute Ranking

This is one of the simplest (and fastest) attribute ranking methods and is often used in text categorization applications where the sheer dimensionality of the data precludes more sophisticated attribute selection techniques [36].

If A is an attribute and C is the class, following equations given the entropy of the class before and after observing the attribute.

$H(C) = -\sum p(c) \log_2(c)$,

$H(C|A) = -\sum P(a) \sum P(c|a) \log_2 P(c|a)$

The amount by which the entropy of the class decreases reflects the additional information about the class provided by the attribute and is called information gain. Each attributes A_i itself and the class:

$$IG_i = H(C) - H(C|A_i)$$
$$= H(A_i) - H(A_i|C)$$
$$= H(A_i) + H(C) - H(A_iC)$$

Data sets with numeric attributes are first discretized using the method of Fayyad and Irani.

II. LITERATURE SURVEY

This section reviews the current literature and related work in the areas of intrusion detection systems concerning with different methods and technology through examination of various research papers, journals and online resources.

Aruna Jamdagni et al. [1] proposed RePIDS and evaluated using DARPA 99 dataset and Georgia Institute of Technology attack dataset. The traffic for Web-based application is considered for validating our model. F-value a criterion is used to evaluate the detection performance of RePIDS. Experimental results show

that RePIDS achieves better performance (high F-values, 0.9958 for DARPA 99 dataset and 0.976 for Georgia Institute of Technology attack dataset respectively, with only 0.85% false alarm rate) and lower computational complexity when compared against two state-of-the-art payload-based intrusion detection systems. Additionally, it has 1.3 time higher throughput in comparison with real scenario of medium sized enterprise network.

Ahmed Patel et al. [2] the latest developed Intrusion Detection Prevention System (IDPSs) and alarm management techniques by providing a comprehensive taxonomy and investigating possible solutions to detect and prevent intrusions in cloud computing systems. The desired characteristics of IDPS and cloud computing systems, a list of germane requirements are identified and four concepts of autonomic computing fuzzy theory, self-management risk management and ontology are leveraged to satisfy these requirements.

Chung-Ming Ou [8] used Agent-based artificial immune system (ABAIS) is adapted to intrusion detection system (IDS). An Agent - Based IDS (ABIDS) inspired by the danger theory of human immune system is proposed. Multiple agents are entrenched to ABIDS where agents coordinate one another to calculate Mature Context Antigen Value (MCAV) and update activation threshold for security responses. The intelligence behind ABIDS is based on the danger theory and the functionalities of dendritic cells in human immune systems, while Dendritic Cells agents (DC agent) are emulated for innate immune subsystem and artificial T-Cell agents (TC agent) are for adaptive protected subsystem. Antigens are profiles of system calls while corresponding behaviours are regarded as signals. This ABIDS is based on the dual detections of DC agents for signals and TC agents for antigens. ABAIS is an intelligent system with learning technique and memory capabilities. According to MCAVs immune response to malicious behaviours is activated by either computer host or Security Operating Centre. Accordingly computer hosts met with malicious intrusions can be effectively detected by input signals and temporary output signals such as PAMP danger and safe signals.

Chenfeng Vincent Zhou et al. [9] summarized the current research directions in detecting such attacks using collaborative intrusion detection systems (CIDSs). In particular highlight two main challenges in CIDS research. CIDS architectures and alert correlation algorithms. In this paper review the current CIDS approaches in terms of these two challenges conclude by highlighting opportunities for an integrated solution to large-scale collaborative intrusion detection.

C. Kolias et al. [10] explored the reasons that led to the application of Swarm Intelligence (SI) in intrusion detection and present SI methods that have been used for constructing IDS. A main contribution is also a detailed comparison of several SI-based IDS in terms of efficiency. This gives a clear idea of which solution is more appropriate for each particular case.

Dr. Saurabh and Neelam [11] suggested identifying important reduced input features in building IDS that is computationally efficient and effective. This paper investigates the performance of three standard feature selection method using correlation-based Feature Selection, Information Gain and Gain Ratio. In this paper propose method Feature Vitality Based Reduction Method, to identify important reduced input features.

D. Mutz et al. [13] argue that most hybrid systems obtain high false alarm rates due to simplistic approaches to combining the outputs of the techniques in the decision phase. They propose a hybrid host based anomaly detection system consisting of four detection techniques: analysing string length, character distribution, and structure, and identifying learned tokens, in which a Bayesian network is employed to decide the final output classification. The system was validated on the DARPA99 dataset, compared with a simple threshold based approach. Both approaches (Bayesian and threshold) were given the same outputs from the detection techniques. With 90% true positives, the threshold based approach lead to twice as many false positives as the Bayesian network.

Guorui Li et al. [17] proposed a distributed group-based intrusion detection scheme that meets all the above requirements by partitioning the sensor networks into many groups in which the sensors in each group are physically close to each other and are equipped with the same sensing capability. Intrusion detection algorithm takes simultaneously into consideration of multiple attributes of the sensor nodes to detect malicious attackers precisely. In this paper show through experiments with real data that our algorithm can decrease the false alarm rate and increase the detection accuracy compared with existing intrusion detection schemes while lowering the computation and transmission power consumption.

Hung-Jen Liao et al. [18] proposed declared that an Intrusion Detection System (IDSs) has received a lot of attention throughout the computer science field. Existing IDSs pose challenges on not only capricious intrusion categories, but also huge computational authority. Though there is a number of existing literatures to IDS issues, in this paper show attempt to give a more elaborate image for a comprehensive review. Through the extensive survey and sophisticated organization, propose the taxonomy to outline modern IDSs.

Levent Koc et al. [27] used technique such as pattern recognition and the data mining of network events are often used by intrusion detection system to classify the network event as either normal events or attack events. In this research paper study claims that the Hidden Naive Bayes (HNB) model can be applied to intrusion detection problems that suffer from dimensionality extremely correlated features and high network data stream volumes. HNB is a data mining model that relaxes the Naive Bayes method's conditional independence assumption. This paper experimental result show that the HBN model exhibits a superior overall

performance in terms of accuracy, error rate and misclassification cast compared with the traditional Naive Bayes model, leading extended Naive Bayes model and the Knowledge Discovery and Data Mining (KDD) cup 1999 winner.

M.Sabhnani and Serpen [29] have examined the performance of several machine learning techniques on the KDD Cup 99 dataset, including a C4.5 DT. The DT obtained good accuracy, but does not perform as well as other techniques on some classes of intrusion, particularly U2R and R2L attacks, both of which are minor classes and include a large proportion of new attack types. An ANN and K-Means clustering obtained higher detection rates on these classes, which are two techniques that are better able to generalize from learned data to new, unseen, data.

Manasi Gyanchandani et al. [36] in this paper evaluates the performance of C4.5 classifier and its combination using bagging, boosting and stacking over NSL KDD dataset for Intrusion Detection System. This dataset usual consists of selected records of the complete KDD dataset.

N. Ben Amor et al. [37] conducted an empirical investigation on the KDD Cup 99 dataset, comparing the performance of NB and a Decision Tree (DT). The DT obtains a higher accuracy (92.28% compared with 91.47%), but NB obtains better detection rates on the three minor classes1, namely Probing, U2R and R2L intrusions. Most significantly, the DT detects merely 0.52% R2L intrusions whilst NB detects 7.11%.

P. Garcia Teodoro et al. [39] described security tools incorporating anomaly detection functionalities are just starting to appear, and several importance problems remain to be solved. This paper in the most well-known anomaly-based intrusion detection techniques platforms systems under development and research projects in the area presented. Finally the main challenges to be dealt with for the wide scale deployment of anomaly-based intrusion detectors with special emphasis on assessment issues.

Phurivite Sangkatsanee et al. [40] proposed a real-time intrusion detection approach using a supervised machine learning technique. Authors approach is simple and efficient and can be used with many machine learning techniques. In this paper used applied different well-known machine learning techniques to evaluate the performance of IDS approach and experimental result show that the Decision Tree technique can outperform the other techniques. This research paper also identified 12 essential features of network data which are relevant to detecting network attack using the information gain as feature selection criterions and developed a new post-processing processing procedure to reduce the false-alarm rate as well as increase the reliability and detection accuracy of the intrusion detection system.

Simon T. Powers et al. [41] evaluated a hybrid system specifically anomalous network connections are initially detected using artificial immune system connections that are flagged as anomalous are then categorised using a Kohonen Self Organising Map allowing higher-level information in the form of cluster membership to be removed. Experimental results on the KDD 1999 cup dataset show a low false positive rate and a detection and classification rate for Denial of Service (DoC) and User to Root (U2R) attacks that is higher than those in a sample of other works.

Sanjay Rawat et al. [42] presented study of investigate the applicability of Spectral Analysis technique Singular Value Decomposition (SVD) as a pre-processing step to reduce the dimensionality of the data. This reduction highlights the most prominent features in the data by eliminating the noise. This pre-processing step not only makes the data noise-free but also reduces the dimensionality of the data thereby minimizing computational time. This research paper proposed technique can be applied to other existing methods to improve their performance. Perform experiments on various data sets like DARPA 98, UNM send mail, inetd, and login-ps data sets to show that reduction in the dimension of the data does not degrade the performance of the IDS. In fact in case of single application observing like send mail, by applying reduction techniques get very encouraging results.

S. Peddabachigari et al. [43] conducted an empirical investigation of SVMs and DTs, in which they analyzed their performance as standalone detectors and as hybrids. In this paper two hybrid models were examined a hierarchical model (DT-SVM) with the DT as the first layer, to produce node information for the SVM in the second layer, and an ensemble model comprising the standalone techniques and the hierarchical hybrid. For the ensemble approach each technique is given a weight according to detection rate of each particular attack type during training.

Yuk Ying Chung et al. [45] proposed a new hybrid intrusion detection system by using Intelligent Dynamic Swarm based Rough Set (IDS-RS) for feature selection and simplified swarm optimization for intrusion data classification. It is proposed to select the most relevant features that can represent the pattern of the network traffic. Improve the performance of SSO classifier a new Weighted Local Search (WLS) strategy incorporated in Simplified Swarm Optimization (SSO)is proposed. The purpose of this new local search strategy is to discover the better solution from the neighbourhood of the current solution produced by SSO. The performance of the proposed hybrid system on KDD Cup 99 dataset has been evaluated by comparing it with the standard Particle Swarm Optimization (PSO) and two other most popular standard classifiers. The testing results showed that the proposed hybrid system can achieve higher classification accuracy than others with 93.3% and it can be one of the competitive classifier for the intrusion detection system.

III. PROPOSED METHODOLOGY

In intrusion detection system, it is essential to perform better for unknown attack. This work evaluates the performance of various classification algorithms for the test dataset of novelty attacks as well as on the Original and Prepossessed test dataset.

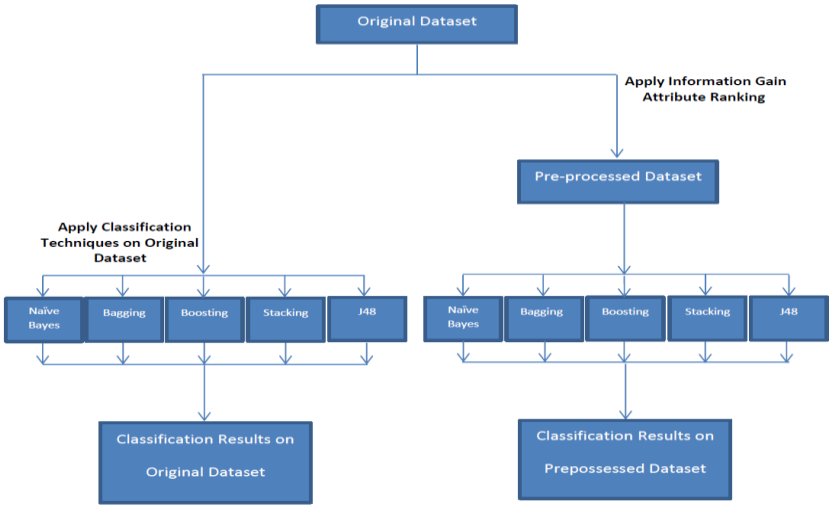

Figure 3.1 Proposed Models for Feature Extraction based Classification Techniques

3.1 Proposed Algorithm

Basically the four steps are used in this framework which is given below:

Step1: Apply classification techniques using Naïve Bayes, Bagging, Boosting, Stacking and J48 on Dataset DS0 for categorizing of different network attacks namely Normal, Probe, DoS, U2R and R2L.

Step2: Apply feature extraction through Information Gain Attribute Ranking technique to reduce the dimension of Original Dataset DS0 for preprocessing which produce Dataset DS1.

Step3: Apply classification techniques using Naïve Bayes, Bagging, Boosting, Stacking and J48 on Dataset DS1 for categorizing of different network attacks namely Normal, Probe, DoS, U2R and R2L.

Step4: Comparison of classification results between Original and Preprocessed dataset.

IV. EXPERIMENTAL SETUP

4.1 Experiment Design

Classifiers used for the experiments are Naïve Bayes, Bagging, Boosting, Stacking, and J48. Two dataset created from NSL-KDD dataset are used as input. To conduct experiments a WEKA tool is used which contains implementation of various machine learning algorithms used for data mining. RunWEKA.ini file is edited to assign 1.5 GB of memory to WEKA in order to handle large volume of data.

4.2 Evaluation metrics

Metrics which are mainly used to evaluate the performance of classifier are present in [38].

The **True Positives** (TP) and **True Negatives** (TN) are correct classifications.

A **False Positive** (FP) occurs when the outcome is incorrectly predicted as yes (or positive) when it is actually no (negative).

A False Negative (FN) occurs when the outcome is incorrectly predicted as negative when it is actually positive.

Probability of Detection (PD)/Recall: The percentage of the total relevant documents in a database retrieved by the search. If you knew that there were 1000 relevant documents in a database and your search retrieved 100 of these relevant documents, your recall would be 10%.

$$\textbf{PD/Recall} = \frac{\text{Total\_Detected\_Attacked}}{\text{Total\_Attacks}} * 100$$

$$\textbf{Recall} = TP/(TP+FN)$$

False Alarm Rate (FAR): The percentage of false alarms given the event did not occurred.

$$\textbf{FAR} = \frac{\text{Total\_Misclassified\_Process}}{\text{Total\_Normal\_Process}} * 100$$

$$\textbf{False Alarm rate} = FP/(TN+FN)$$

Precision: The percentage of relevant documents in relation to the number of documents retrieved. If your search retrieves 100 documents and 20 of these are relevant, your precision is 20%.
Precision = TP/ (TP+FP)
F-measure: The harmonic mean of precision and recall
F = 2 * Recall * Precision / (Recall + Precision)
The **True Positive rate** is TP divided by the total number of positives, which is TP + FN.
The **False Positive rate** is FP divided by the total number of negatives, FP + TN.

		Actual Result	
		Intrusion	Normal
Predicted Result	Intrusion	True Positive (TP)	False Positive (FP)
	Normal	False Negative (FN)	True Negative (TN)

Fig 4.1 Predicted Classes

4.3 Preprocessing of data

It has been found that model generation is computation intensive. So in order to reduce time redundant attributes can be removed, (which may also insert noise in the task of classification) by various feature selection algorithms. In this work summarizes the feature selection algorithms and the search method used to generate dataset DS1 from original dataset.

Dataset	Feature Selection Algorithms	Search Method	No. of Attributes
Ds0	Original dataset	None	42
Ds1	InfoGainAttributeEval	Ranker	24

Table 4.1 Data set Generation

V. RESULTS

The comparative results analysis of Classification Techniques using F-Measure is given in this section:

5.1 Naïve Bayes Classifiers for DS0

Class	Precision	Recall	F-Measure
Normal	**0.71**	**0.97**	**0.81**
Probe	**0.71**	**0.63**	**0.64**
DoS	**0.98**	**0.96**	**0.97**
U2R	**0.32**	**0.43**	**0.24**
R2L	**0.43**	**0.45**	**0.65**

Table 5.1 Results of Naïve Bayes Classifiers for DS0

Table 5.1 shows that Result of Naïve Bayes classifier detects the highest possibility of DoS attack and lowest possibility of U2R attack, displayed using bar chart in Figure 5.1.

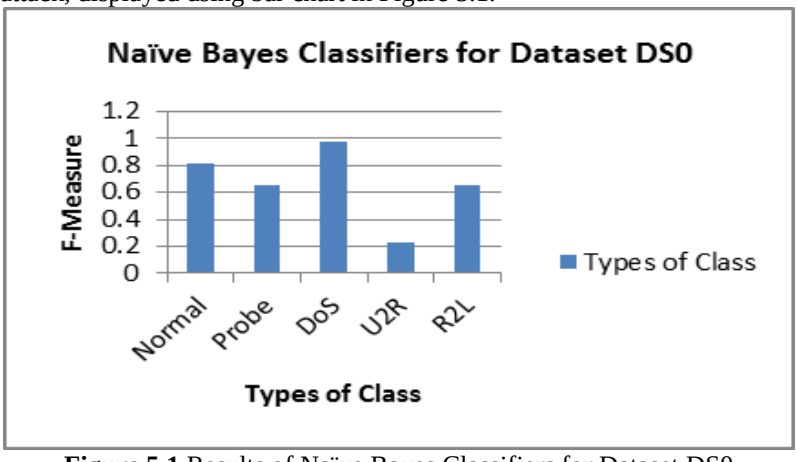

Figure 5.1 Results of Naïve Bayes Classifiers for Dataset DS0

5.2 Naïve Bayes Classifiers for DS1

Class	Precision	Recall	F-Measure
Normal	**0.78**	**0.77**	**0.78**
Probe	**0.68**	**0.28**	**0.92**
DoS	**0.89**	**0.97**	**0.94**
U2R	**0.66**	**0.18**	**0.45**
R2L	**0.07**	**0.87**	**0.54**

Table 5.2 Results of Naïve Bayes Classifiers for DS1

Table 5.2 shows that Result Naïve Bayes classifier detects the highest possibility of DoS attack and lowest possibility of U2R attack, displayed using bar chart in following Figure 5.2.

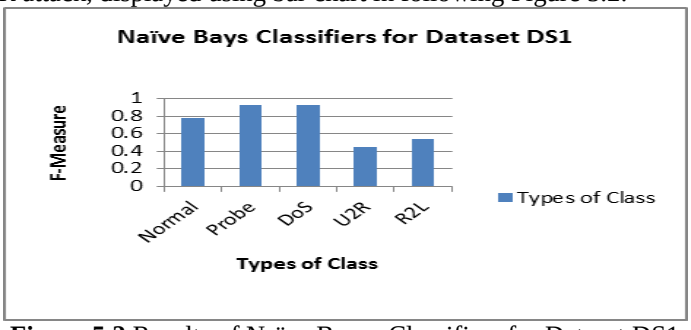

Figure 5.2 Results of Naïve Bayes Classifiers for Dataset DS1

5.3 Bagging Classifiers for DS0

Class	Precision	Recall	F-Measure
Normal	**0.73**	**0.99**	**0.84**
Probe	**0.92**	**0.74**	**0.82**
DoS	**0.99**	**0.97**	**0.95**
U2R	**0.68**	**0.57**	**0.25**
R2L	**0.98**	**0.56**	**0.23**

Table 5.3 Results of Bagging Classifiers for DS0

Table 5.3 shows that Result of Bagging classifier detects the highest possibility of DoS attack and lowest possibility of R2L attack, displayed using bar chart in following Figure 5.3.

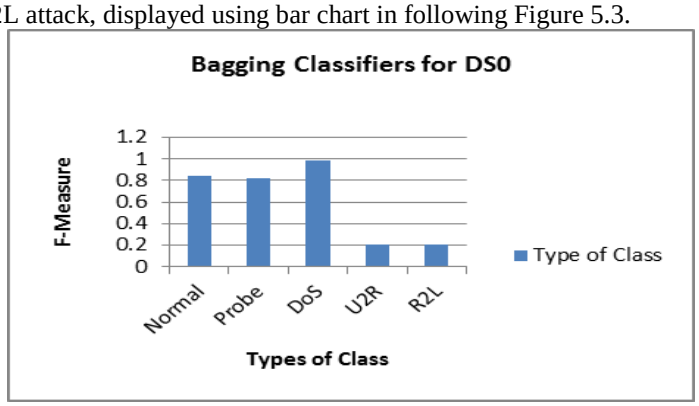

Figure 5.3 Results of Bagging Classifiers for Dataset DS0

5.4 Bagging Classifiers for DS1

Class	Precision	Recall	F-Measure
Normal	**0.75**	**0.97**	**0.83**
Probe	**0.73**	**0.64**	**0.82**
DoS	**0.98**	**0.97**	**0.96**
U2R	**0.18**	**0.18**	**0.25**
R2L	**0.14**	**0.79**	**0.23**

Table 5.4 Results of Bagging Classifiers for DS1

Table 5.4 shows that Result of Bagging classifier detects the highest possibility of DoS attack and lowest possibility of R2L attack, displayed using bar chart in following Figure 5.4.

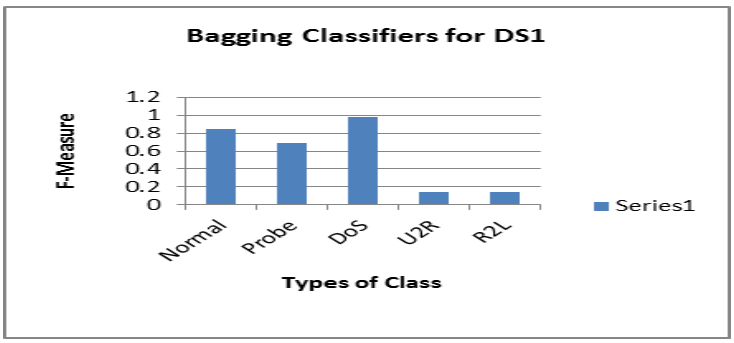

Figure 5.4 Results of Bagging Classifiers for Dataset DS1

5.5 Boosting Classifiers for DS0

Class	Precision	Recall	F-Measure
Normal	**0.75**	**0.91**	**0.87**
Probe	**0.73**	**0.74**	**0.74**
DoS	**0.95**	**0.96**	**0.96**
U2R	**0.45**	**0.13**	**0.26**
R2L	**0.84**	**0.16**	**0.41**

Table 5.5 Results of Boosting Classifiers for DS0

Table 5.5 shows that Result of Boosting classifier detects the highest possibility of DoS attack and lowest possibility of U2R attack, displayed using bar chart in following Figure 5.5.

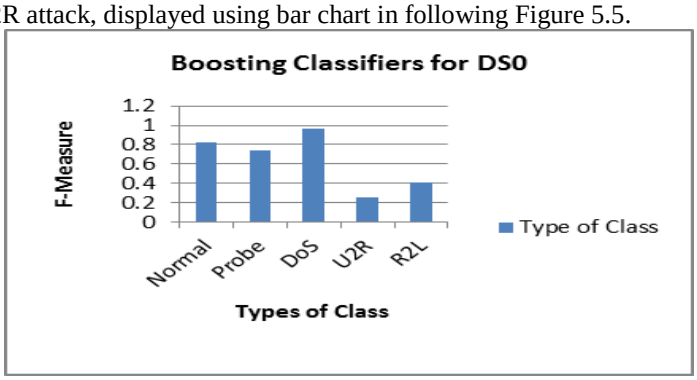

Figure 5.5 Results of Boosting Classifiers for Dataset DS0

5.6 Boosting Classifiers for DS1

Class	Precision	Recall	F-Measure
Normal	**0.78**	**0.77**	**0.77**
Probe	**0.68**	**0.28**	**0.39**
DoS	**0.89**	**0.97**	**0.93**
U2R	**0.66**	**0.18**	**0.34**
R2L	**0.75**	**0.71**	**0.66**

Table 5.6 Results of Boosting for DS1

Table 5.6 shows that Result of Boosting classifier detects the highest possibility of DoS attack and lowest possibility of U2R attack, displayed using bar chart in following Figure 5.6.

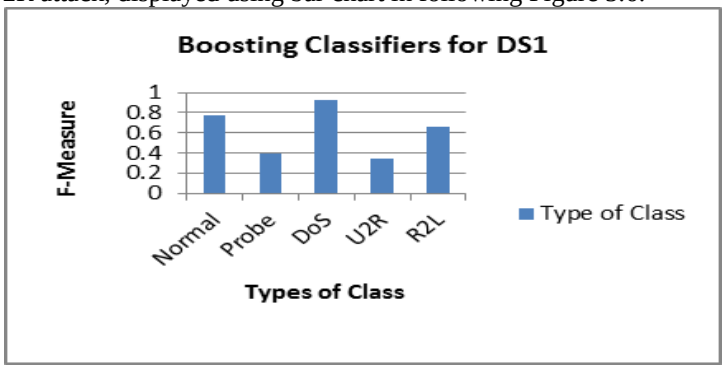

Figure 5.6 Results of Boosting Classifiers for Dataset DS1

5.7 Stacking Classifiers for DS0

Class	Precision	Recall	F-Measure
Normal	**0.73**	**0.98**	**0.83**
Probe	**0.71**	**0.81**	**0.76**
DoS	**0.99**	**0.97**	**0.98**
U2R	**0.52**	**0.16**	**0.34**
R2L	**0.88**	**0.07**	**0.24**

Table 5.7 Results of Stacking Classifiers for DS0

Table 5.7 shows that Result of Stacking classifier detects the highest possibility of DoS attack and lowest possibility of R2L attack, displayed using bar chart in following Figure 5.7.

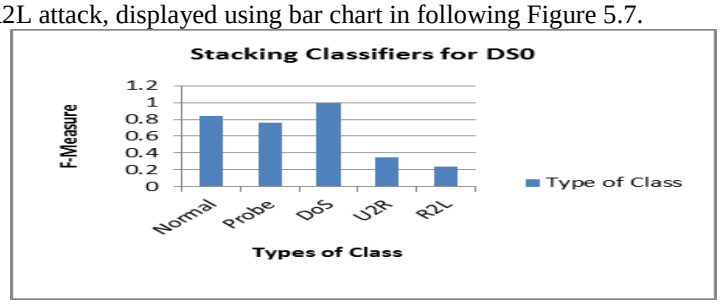

Figure 5.7 Results of Stacking Classifiers for Dataset DS0

5.8 Stacking Classifiers for DS1

Class	Precision	Recall	F-Measure
Normal	**0.74**	**0.97**	**0.84**
Probe	**0.71**	**0.63**	**0.67**
DoS	**0.98**	**0.97**	**0.98**
U2R	**0.75**	**0.71**	**0.75**
R2L	**0.68**	**0.43**	**0.81**

Table 5.8 Results of Stacking Classifiers for DS1

Table 5.8 shows that Result of Stacking classifier detects the highest possibility of DoS attack and lowest possibility of Probe attack, displayed using bar chart in Figure 5.8.

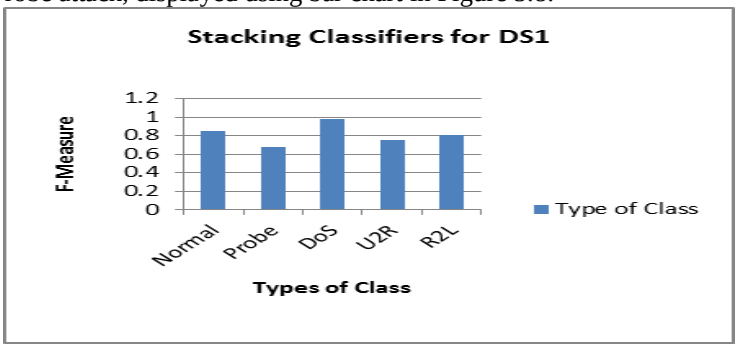

Figure 5.8 Results of Stacking Classifiers for Dataset DS1

5.9 J48 Classifiers for DS0

Class	Precision	Recall	F-Measure
Normal	**0.77**	**0.98**	**0.86**
Probe	**0.45**	**0.56**	**0.57**
DoS	**0.61**	**0.54**	**0.99**
U2R	**0.62**	**0.16**	**0.18**
R2L	**0.49**	**0.57**	**0.76**

Table 5.9 Results of J48 Classifiers for DS0

Table 5.9 shows that Result of J48 classifier detects the highest possibility of DoS attack and lowest possibility of U2R attack, displayed using bar chart in following Figure 5.9.

Figure 5.9 Results of J48 Classifiers for Dataset DS0

5. 10 J48 Classifiers for DS1

Class	Precision	Recall	F-Measure
Normal	**0.81**	**0.97**	**0.88**
Probe	**0.49**	**0.53**	**0.51**
DoS	**0.12**	**0.55**	**0.75**
U2R	**0.34**	**0.32**	**0.57**
R2L	**0.71**	**0.89**	**0.46**

Table 5.10 Results of J48 Classifiers for DS1

Table 5.10 shows that Result of J48 classifier detects the highest possibility of Normal attack and lowest possibility of R2L attack, displayed using bar chart in following Figure 5.10.

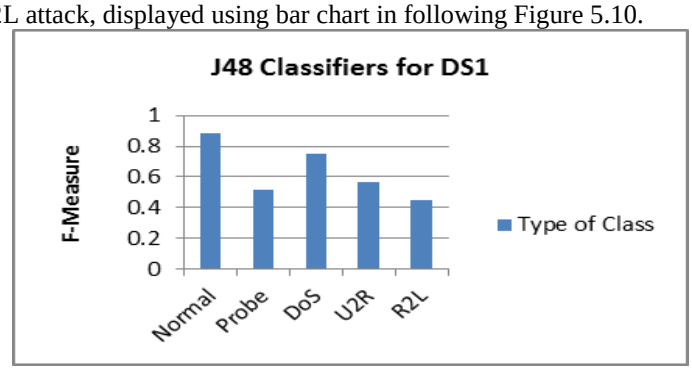

Figure 5.10 Results of J48 Classifiers for Dataset DS1

5.11 Comparative Analysis for DS0

Class	Naïve Bayes	Bagging	Boosting	Stacking	J48
Normal	**0.81**	**0.84**	**0.87**	**0.83**	**0.86**
Probe	**0.64**	**0.82**	**0.74**	**0.76**	**0.57**
DoS	**0.97**	**0.95**	**0.96**	**0.98**	**0.99**
U2R	**0.24**	**0.25**	**0.26**	**0.34**	**0.18**
R2L	**0.65**	**0.23**	**0.41**	**0.24**	**0.76**

Table 5.11 Result of Performance evaluation (F-Measure) of difference Classifiers for DS0

Table 5.11 shows that performance evaluation, J48 classifier detects the highest possibility of DoS attack and Bagging Classifier shows lowest possibility of R2L attack for Original Dataset DS0, displayed using bar chart following Figure 5.11.

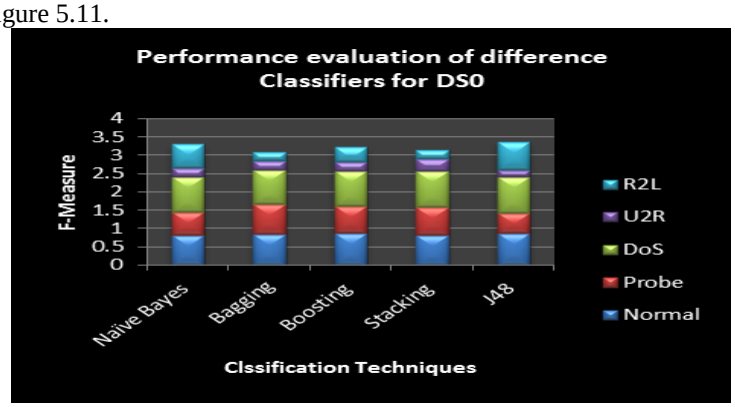

Figure 5.11 Result of Performance evaluation (F-Measure) of difference Classifiers for DS0

5.12 Comparative Analysis for DS1

Class	Naïve Bayes	Bagging	Boosting	Stacking	J48
Normal	**0.78**	**0.83**	**0.77**	**0.84**	**0.88**
Probe	**0.92**	**0.82**	**0.39**	**0.67**	**0.51**
DoS	**0.94**	**0.96**	**0.93**	**0.98**	**0.75**
U2R	**0.45**	**0.25**	**0.34**	**0.75**	**0.57**
R2L	**0.54**	**0.23**	**0.66**	**0.81**	**0.46**

Table 5.12 Result of Performance evaluation (F-Measure) of difference Classifiers for DS1

Table 5.12 shows that performance evaluation, Stacking classifier detects the highest possibility of DoS attack and Bagging Classifier shows lowest possibility of R2L attack for Preprocessed Dataset DS1, displayed using bar chart following Figure 5.12.

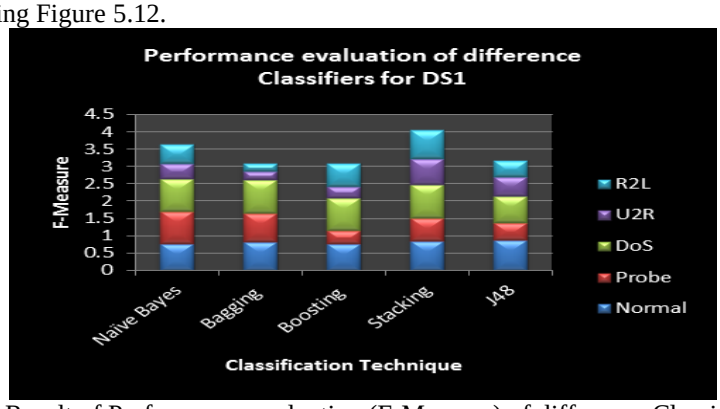

Figure 5.12 Result of Performance evaluation (F-Measure) of difference Classifiers for DS1

The conclusion and feature scope of discussed in the next.

VI. CONCLUSION AND FUTURE SCOPE

This research is approached to discover the best performance of classification algorithm for intrusion detection. The evaluation of two types of dataset Original and Prepossessed with the different network attacks namely Normal, Probe, DoS, U2R and R2L. Prepossessed dataset obtained to reduction of the features using information gain technique. The experiment results show that J48 classifier detects the highest possibility of DoS attack and Bagging Classifier shows lowest possibility of R2L attack for Original Dataset DS0, Whereas Stacking classifier detects the highest possibility of DoS attack and Bagging Classifier shows lowest possibility of R2L attack for Preprocessed Dataset DS1.

In the present study few issues like high dimensionality, Scalability and accuracy are focused but there are still many issues that can be taken into consideration for further research which are as different algorithms which are not included in WEKA can be tested. Also, experiments with various feature selection techniques should be compared. Classification technique of data mining is useful in every domain of our life e.g. University domain category wise, Medical domain, crime domain, Auto Price, Zoo etc. Cost based classifier can be applied to IDS which keeps track of cost matrix which contains cost of misclassification. Classifier combination which has Trees based classifier. As demonstrated in the result section Trees gives best precision (equal to one) for the normal class the packets classified as normal are declared normal. For the rest of instances i.e. instances classified as attack we can use some good classifier at another level.

REFERENCES

[1] Aruna Jamdagni, Zhiyuan Tan, Xiangjian He, Priyadarsi Nanda, Ren Ping Liu, RePIDS: A multi-tier Real-time Payload-based Intrusion Detection System Computer Networks pp. 01–14, 2012

[2] Ahmed Patel, MonaTaghavi, KavehBakhtiyari, Joaquim, Celestino Ju, nior, An intrusion detection and prevention system in cloud computing: A systematic review, Journal of Network and Computer Applications pp. 25–41, 2013

[3] A.N. Toosi, M. Kahani, A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers, Computer Communications pp. 2201–2212, 2007.

[4] A. Tajbakhsh, M. Rahmati, A. Mirzaei, Intrusion detection using fuzzy association rules, Applied Soft Computing pp. 462–469, 2009.

[5] Abdoul Karim Ganame RB, Bourgeoisa Julien, Spiesa F. A global security architecture for intrusion detection on computer networks. Computers & Security pp. 30–47, 2008.

[6] Almgren M, Lindqvist U, Jonsson E. A multi-sensor model to improve automated attack detection. In: Proceedings of the 11th international symposium on recent advances in intrusion detection, pp. 291–310, 2008.

[7] Balasubramaniyan J, Garcia-Fernandez J, Isacoff D, Spafford E, Zamboni D. An architecture for intrusion detection using autonomous agents. In: Proceedings of the 14th IEEE computer security applications conference pp. 13–24, 1998.

[8] Chung-Ming Ou, Host-based intrusion detection systems adapted from agent-based artificial immune systems, Neurocomputing pp. 78–86, 2012.

[9] Chenfeng Vincent Zhou, Christopher Leckie, Shanika Karunasekera, A survey of coordinated attacks and collaborative intrusion detection, computers & security 29, pp. 124–140, 2010.

[10] C. Kolias, G. Kambourakis, M. Maragoudakis, Swarm intelligence in intrusion detection: A survey, computers & security 30, pp. 625-642, 2011.

[11] Dr. Saurabh Mukherjee, Neelam Sharma, Intrusion Detection using Naive Bayes Classifier with Feature Reduction, Procedia Technology 4, pp. 119 – 128, 2012.

[12] D. Dasgupta, F. Gonzalez, K. Yallapu, J. Gomez, R. Yarramsettii, CIDS: An agent-based intrusion detection system, Computers & Security 24, pp. 387-398, 2005.

[13] D. Mutz, F. Valeur, G. Vigna and C. Kruegel. Anomalous system call detection. ACM Trans. Inf. Syst. Secur., volume 9, ISSN 1094-9224, pp. 61-93, 2006.

[14] Fenet S, Hassas S. A distributed intrusion detection and response system based on mobile autonomous agents using social insects communication paradigm. In: Proceedings of the First International Workshop on Security of Mobile Multiagent Systems (SEMAS), pp. 41-58, 2001.

[15] Feng Y, Wu ZF, Wu KG, Xiong ZY, Zhou Y. An unsupervised anomaly intrusion detection algorithm based on swarm intelligence. In: the Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, pp. 3965-3969, 2005.

[16] Feng Y, Zhong J, Ye CY, Wu ZF. Clustering based on self-organizing ant colony networks with application to intrusion detection. In: Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications (ISDA '06), pp. 1077-1080, 2006.

[17] Guorui Li, Jingsha He, Yingfang Fu, Group-based intrusion detection system in wireless sensor networks, Computer Communications 31, pp. 4324–4332, 2008.

[18] Hung-Jen Liao, Chun-HungRichardLin, Ying-ChihLin, Kuang-YuanTung, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications 36, pp. 16–24, 2011.

[19] H.G. Kayacik, A.N. Zincir-Heywood, M.I. Heywood, on the capability of an SOM based intrusion detection system, in: Proceedings of the 2003 International Joint Conference on Neural Networks, vol. 3, IEEE Press, pp. 432-444, 2003.

[20] H.A. Nguyen, D. Choi, Application of data mining to network intrusion detection: classifier selection model, in: LNCS, vol. 5297, Springer-Verlag, Berlin Heidelberg, pp. 399–408, 2008.

[21] Heberlein LT, Mukherjee B, Levitt KN. Internetwork security monitor: an intrusion-detection system for large-scale networks. In: Proceedings of the 15th national computer security conference, pp. 262–271, 1992.

[22] Hochberg J, Jackson K, Stallings C, McClary JF, DuBois D, Ford J. Nadir: an automated system for detecting network intrusion and misuse. In: Proceedings of the 15th national computer security conference, pp. 235–48, 1993.

[23] Huang M, Jasper R, Wicks T. A large scale distributed intrusion detection framework based on attack strategy analysis. Computer Networks, pp. 2465–2475, 1999.

[24] J. Kim, P.J. Bentley, Evaluating negative selection in an Artificial Immune System for network intrusion detection, in: Proceedings of the 2001 Genetic and Evolutionary Computation Conference (GECCO'01), Morgan Kaufmann, pp. 387-399, 2001.

[25] J.H. Lee, J.H. Lee, S.G. Sohn, J.H. Ryu, T.H. Chung, Effective value of decision tree with KDD 99 intrusion detection datasets for intrusion detection system, in: International Conference on Advanced Communication Technology (ICACT), Korea, pp. 789-799, 2008.

[26] K.P. Anchor, P. Williams, G. Gunsch, G. Lamont, The computer defense immune system: current and future research in intrusion detection, in: D.B. Fogel, M.A. El- Sharkawi, X. 12–17 May 2002, IEEE Press, pp. 1027–1032, 2002.

[27] Levent Koc, Thomas A. Mazzuchi, Shahram Sarkani, A network intrusion detection system based on a Hidden Naïve Bayes multiclass classifier, Expert Systems with Applications 39, pp. 13492–13500, 2012.

[28] L.L. DeLooze, Attack characterization and intrusion detection using an ensemble of self-organizing maps, in: Proceedings of the 2006 IEEE Information Assurance Workshop, IEEE Press, pp. 897-915, 2006.

[29] M. Sabhnani and G. Serpen, Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context. In Proceedings of the International Conference on Machine Learning, Models, Technologies and Applications (MLMTA 2003), volume 1, pp. 209-215, 2003.

[30] M. Asaka, T. Onabura, T. Inoue, S. Okazawa, S. Goto, A new intrusion detection method based on discriminant analysis, IEICE Transactions on Information and System 5 (May), pp. 570–577, 2001.

[31] Moore D, Shannon C, Brown J. Code Red: a case study on the spread and victims of an Internet worm. In: Proceedings of the 2002 ACM SICGOMM Internet Measurement Workshop, pp.273–84, 2002.

[32] Moore D, Paxson V, Savage S, Shannon C, Staniford S, Weaver N. Inside the slammer worm. IEEE Security & Privacy Magazine, pp.3–9, 2003.

[33] Morin B, Me L, Debar H, Ducasse M. M2D2: a formal data model for IDS alert correlation. In: Proceedings of the 5th international symposium on recent advances in intrusion detection (RAID), pp. 115–37, 2002.

[34] Mounji A, Le Charlier B, Zampunieris D, Habra N. Distributed audit trail analysis. In: Proceedings of the internet society symposium on network and distributed system security (ISOC), pp. 102–13, 1995.

[35] M. Ayara, J. Timmis, R. de Lemos, L.N. de Castro,, R. Duncan, Negative selection: how to generate detectors, in: J. Timmis, P.J. Bentley (Eds.), Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS'02), pp. 89–98, 2002.

[36] Manasi Gyanchandani, R. N. Yadav, J. L. Rana, Intrusion Detection using C4.5: Performance Enhancement by Classifier Combination, ACEEE Int. J. on Signal & Image Processing, Vol. 01, No. 03, pp. 46-49, 2010.

[37] N. Ben Amor, S. Benferhat and Z. Elouedi. Naive Bayes vs Decision Trees in Intrusion Detection Systems. In SAC ‟04: Proceedings of the 2004 ACM symposium on Applied computing, New York, NY, USA. ACM. ISBN 1-58113-812-1, pp. 420-424, 2004.

[38] O. Depren, M. Topallar, E. Anarim, M.K. Ciliz, An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, Expert Systems with Applications 29, pp. 713–722, 2005.

[39] P. Garcı´a-Teodoro, J. Dı´az-Verdejo, G. Macia´-Ferna´ndez, E. Va´zquez Anomaly-based network intrusion detection: Techniques, systems and challenges, computers & security 28, pp. 18–28, 2009.

[40] Phurivit Sangkatsanee, Naruemon Wattanapongsakorn, Chalermpol Charnsripinyo, Practical real-time intrusion detection using machine learning approaches, Computer Communications 34, pp. 2227–2235, 2011.

[41] Simon T. Powers, Jun He, A hybrid artificial immune system and Self Organising Map for network intrusion detection, Information Sciences 178, pp. 3024–3042, 2008.

[42] Sanjay Rawat, Arun K. Pujari, V. P. Gulati, On the Use of Singular Value Decomposition for a Fast Intrusion Detection System, Electronic Notes in Theoretical Computer Science 142, pp. 215–228, 2006.

[43] S. Peddabachigari, A. Abraham, C. Grosan and J. Thomas, Modeling Intrusion Detection Systems Using Hybrid Intelligent Systems. Journal of Network and Computer Applications, 30, pp. 114-132, 2007.

[44] S. Pukkawanna, V. Visoottiviseth, P. Pongpaibool, Lightweight detection of DoS attacks, in: The IEEE International Conference on Networks (ICON), Australia, pp. 77–82, 2007.

[45] Yuk Ying Chunga, Noorhaniz Wahid, A hybrid network intrusion detection system using simplified swarm optimization (SSO), Applied Soft Computing 12, pp. 3014–3022, 2012.