Accepted Manuscript

Title: Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model

Authors: Shadi Aljawarneh, Monther Aldwairi, Muneer Bani Yasin

PII: S1877-7503(16)30509-9
DOI: http://dx.doi.org/doi:10.1016/j.jocs.2017.03.006
Reference: JOCS 628

To appear in:

Received date: 24-12-2016
Revised date: 5-2-2017
Accepted date: 5-3-2017

Please cite this article as: Shadi Aljawarneh, Monther Aldwairi, Muneer Bani Yasin, Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model, Journal of Computational Sciencehttp://dx.doi.org/10.1016/j.jocs.2017.03.006

Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model

Shadi Aljawarneh[1], Monther Aldwairi[12], Muneer Bani Yasin[1]
saaljawarneh@just.edu.jo, monther.aldwairi@zu.ac.ae, masadeh@just.edu.jo
[1]Faculty of Computer and Information Technology, Jordan University of Science and Technology, Irbid, Jordan
[2]College of Technological Innovation, Zayed University

*Corresponding author information:
Shadi Aljawarneh
Email: **saaljawarneh@just.edu.jo**
Phone: +962 2 7201000

Paper Highlights

- Utilise the NSL-KDD data set and the binary and multiclass problem with a 20% training dataset.
- This paper studied a new model that can be used to estimate the intrusion scope threshold degree based on the network transaction data's optimal features that were made available for training.
- The experimental result revealed that the hybrid approach had a significant effect on the minimisation of the computational and time complexity involved when determining the feature association impact scale. The accuracy of the proposed model was satisfactory at 99.77% and 99.63% for the binary class and multiclass NSL-KDD data sets, respectively.

Abstract

Efficiently detecting network intrusions requires the gathering of sensitive information. This means that one has to collect large amounts of network transactions including high details of recent network transactions. Assessments based on meta-heuristic anomaly are important in the intrusion related network transaction data's exploratory analysis. These assessments are needed to make and deliver predictions related to the intrusion possibility based on the available attribute details that are involved in the network transaction. We were able to utilize the NSL-KDD data set, the binary and multiclass problem with a 20% testing dataset. This paper develops a new hybrid model that can be used to estimate the intrusion scope threshold degree based on the network transaction data's optimal features that were made available for training. The experimental results revealed that the hybrid approach had a significant effect on the minimisation of the computational and time complexity involved when determining the feature association impact scale. The accuracy of the proposed model was measured as 99.81% and 98.56% for the binary class and multiclass NSL-KDD data sets, respectively.
However, there are issues with obtaining high false and low false negative rates. A hybrid approach with two main parts is proposed to address these issues. First, data needs to be filtered using the Vote algorithm with Information Gain that combines the probability distributions of these base learners in order to select the important features that positively affect the accuracy of the proposed model. Next, the hybrid algorithm consists of following classifiers: J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and NaiveBayes. Based on the results obtained using the proposed model, we observe improved accuracy, high false negative rate, and low false positive rule.

Keywords: feature reduction; intrusion detection; correlation analysis; association impact scale

1. Introduction

Intrusion detection systems (IDS) are generally divided into two types (see Figure 1): misuse and anomaly intrusion detection systems. For a misuse IDS, instructions are identified based on parameters of system weaknesses and known attack signatures. However, it does not recognise attacks that are new or unfamiliar. On the other hand, anomaly IDS is based on normal behaviour parameters and utilizes them to pinpoint any action that deviates significantly from normal behaviour. The misuse intrusion detection mechanism identifies intrusions by matching existing intrusion patterns in consideration for examination with previously identified patterns. On the other hand, anomaly intrusion detection identifies patterns based on the examination of data taken from normal usage [1].

Valuable information is always attractive to attackers and therefore vulnerable to concentrated network attacks. Intrusion refers to the process when an attacker enters the system or system server forwarding malicious packets to the user system so that it can steal, modify, or corrupt any confidential or important information. An attack refers to the illegal sending of network packets through the network. The intrusion can take place over the server or system as a result of existing system vulnerabilities, such as user misuse, system misconfiguration, or program defects. One can also make an intelligent intrusion by putting together multiple vulnerabilities. In a global network, large number of online services and millions of big servers are running in the system. At the same time, such networks become more attractive to more attackers and thus require intelligent intrusion detection models to defend their network system [3, 4, 42].

The following steps are part of an intelligent intrusion or system attack [3]:

- Collecting information: Gathering information about the target involves obtaining all the details and knowledge about the user who will be under attack. This is made possible by executing queries through the use of network commands such as “nslookup”, “whois” to obtain domain name, IP addresses, and server name, etc.
- Probing and scanning: Involves scanning of the target host and checking the system’s unguarded or unprotected areas as it searches for the sensitive information.
- Remote to local access: Refers to the process of gaining user system access by R2L (remote to local) attack types, such as password guessing, buffer overflow attack, and network sniffing. In other words, in an R2L attack, an unknown person sends the network packet in order to gain local access to the user machine and be able to execute commands on the target. This type of attack can be performed by using open ports found on the target machine, utilizing the system vulnerabilities, password guessing etc.
- User to root access: For this type of attack, system vulnerabilities are used by a normal system user to gain root access to the system. They are quite similar to R2L attacks. However, the attacker here is already a normal machine user and he/she will just try to gain root access to the machine.
- Launch attacks: Finally, actual attacks are launched. Example of these attacks are modifying web pages, stealing confidential information, creating a backdoors for future attacks, or accessing another person’s accounts.

Efficient IDS are normally developed through the utilization of data mining techniques due to the fact that they can excellently detect intrusions and adeptly perform generalisations. However, the implementation and installation of such systems can be naturally complex. The systems’ inherent complications can be categorised into distinct problem sets based on competence, accuracy, and usability parameters [1, 2, 42]. However, IDS designed using data mining techniques and mainly those techniques that have their basis on anomaly detection exhibit a higher percentage of false positive incidents in comparison to previous detection techniques that have their basis on handcrafted signature. Hence, it is difficult for these techniques to process data audit and detect online intrusions. Furthermore, the system's learning process requires large amounts of training data and great complexity compared to current available methodologies.

Therefore, building efficient intrusion detection is vital in the network system's defense and helps in sensing attacks over the network. Therefore, a hybrid classification-based intrusion detection model and a feature selection are proposed. Then, the NSL-KDD data set's dimensions are reduced through the implementation of feature selection. Afterwards, with the application of machine learning approach, an intrusion detection model can be built and used to find system attacks and use the captured data to improve intrusion detection. The proposed model needs feature extraction, dimensionality reduction that can reduce the extracted features, and feature selection. The process of feature extraction involves the utilization of all transformation features, which in turn are made up of a mixture of all the initial features. During the process of feature selection, the classification criteria serve as the basis for the selection of features.

Our work has been organized as follows. The related works are discussed in Section 2. In Section 3, overview of the confusion matrix is drawn to indicate the main elements that should be considered to assess the proposed model usability and accuracy. In Section 4, the important classification techniques are described. Section 5 presents the proposed model and its prototype with details of its phases such as pre-processing, normalization, classifier selections, features selection, and post-processing. Section 6 discusses the results, and finally, Section 7 concludes the paper indicating possible future work.

2. Related work

The first IDS ever recorded was based on research conducted by Dorothy E. Denning under the SRI International [5]. It gave way to the solution known as the intrusion detection expert system. To detect known intrusion types, it implements a dual approach that uses a rule-based expert system. Additionally, it utilizes a statistical anomaly detection component that has its basis on host systems, user profiles, and target systems. Later on, a new version known as the next-generation intrusion detection expert system was released by the same research group [6]. The notion of utilizing anomaly detection for information security became mainstream with the release of DARPA Intrusion Detection Evaluation [7] in 1998 and 1999, along with the MIT. However, [8] demonstrated how DARPA datasets are not suited for simulating actual network systems. This makes it necessary to come up with new datasets for IDS development.

Eduardo DelaHoz et al. [1] came up with a classification approach to detect network anomalies by combining self-organising maps and statistical techniques. Feature selection involves the utilization of Fisher's discriminant ratio and principal component analysis (PCA). Network transactions are then classified as normal or anomalous by using probabilistic self-organising maps and noise removal. Ravale et al. [2] came up with a hybrid technique that uses a combination of data mining approaches. The number of attributes related to every data point is reduced using the K-means clustering algorithm. Additionally, the support vector machine's (SVM) radial basis function (RBF) kernel is utilized for classification. Gaikward et al. [3] came up with a machine learning approach to implement IDS. The feature set dimensions are reduced using the genetic algorithm, and the partial decision tree served as the base classifier in implementing the IDS. Sunil Pawar et al. [4] came up with a genetic algorithm-based network IDS that has chromosomes of varying lengths. A chromosome that possesses relevant features is utilized for rule generation. Each rule's fitness is defined using an effective fitness function. To efficiently detect anomalies, each chromosome contains one or more rules.

Fangjun Kuang et al. [9] combined improved chaotic particle swarm optimisation with kernel PCA (KPCA) to come up with a novel SVM model. KPCA is implemented as the SVM's pre-processor in order to shorten training time and reduce the dimension of feature vectors. Moreover, the researchers proposed an improved chaotic particle swarm optimisation process to help determine if the action is normal or intrusive. Aldwairi et al. [10] used artificial bee colony (ABC) for anomaly intrusion detection. They used classification and regression tree (CART) and Bayesian network and Markov blanket (BNMB) for feature selection. However, the old KDD Cup 99 dataset was used for testing and training.

Iftikhar Ahmad et al. [11] proposed a technique that utilized PCA to select feature subsets based on eigenvalues. The authors implemented genetic principal components instead of simply utilizing a traditional approach towards choosing features with the highest eigenvalues like PCA, to choose the sub-set of SVM and features for classification.

Chun Guo et al. [12] came up with a hybrid learning method called the distance sum-based SVM (DSSVM) to model an effective IDS. In DSSVM, feature dimensions of the cluster centres in the data set and the sum of the distances based on the correlation between each data sample are obtained. The SVM is then utilized as a classifier.

Saurabh Mukherjee et al. [13] came up with a feature vitality-based reduction approach that can identify important features, which can then be utilized to identify anomalies in the selection system. The anomalies in the IDS are then detected using the naive Bayes classifier. A large amount of work is currently being performed in the field of intrusion detection. Most of the work focuses on improving the system's ability to detect attacks and improving the network traffic's speed that can be handled.

Snapp et al. [14] came up with a centralised DIDS model. In this method, the distributed intrusion detection system (DIDS) director is considered a central failure point for this architecture. Crosbie and Spafford [15] came up with a distributed IDS. In this system, communication IDS are made to broadcast activities that have been tagged as malicious among themselves in order to help in intrusion detection. Based on the distributed IDS that made use of artificial immune system (AIS), Hosseinpour et al., [16] came up with a DIDS that has its basis on the AIS that utilizes a central engine. This central engine is synced to all the participating IDS. The central engine also functions as a middle-man between two IDS that want to share a detector record. Afzali and Azmi [17] came up with a multi-agent AIS (MAIS-IDS) approach. Compared to individual works, MAIS-IDS achieve higher recognition accuracy when there is collaboration among virtual machines.

Several machine learning techniques were utilized to develop IDS. A clear discussion of the survey for every technique as well as their pros and cons was given in [18, 19]. Based on these surveys, neural networks were revealed to be a promising machine learning technique that can be used for IDS. A neural network is made up of a collection of actions that be utilized to turn a set of inputs to a collection of searched outputs by utilizing a set of nodes, simple processing units, and connections between them. IDS was developed using multi-layer perceptron based on supervised learning techniques [20] and self-organising maps based on the unsupervised learning technique [21]. Using a neural network is an efficient approach that can be used to improve IDS performance based on the anomaly detection and misuse detection models [22]. To assess the performance of their developed IDS, several researchers utilized different existing datasets [23].

Studies [24] revealed that modern IDS find it difficult to handle high speed network traffic. Researchers [25] have also revealed how attackers can take advantage of this weakness to hide their exploits. They do this by using extraneous information to overload an IDS while they execute an attack.

Sekar et al. [26] developed a new NIDS approach based on concise specifications that can classify normal and abnormal sequences of network packet. However, their only focus was on known attack types. Lu et al. [27] proposed a memory efficient multiple-character-approaching architecture that is applicable and well suited for ASIC implementations. The focus of the researchers was mainly on managing memory, but they were unable to identify anomalous behaviours. Some researchers utilized hardware accelerators to perform the NIDS in order to deal with increasing link speed and higher traffic throughput. Das et al. [28] made use of FPGA-based architecture to detect anomalies in network traffic. A dimensionality reduction approach based on principle component analysis was used to identify outliers. Because of the lack of proper relations, this approach was not able to detect all outliers. Artan et al. [29] also utilized FPGA to improve IDS performance. However, they still had difficulties dealing with novel intrusion identifications and complex state flow scenarios. There were also few attempts to develop IDS based on GPU in [30, 31, 32]. However, they also had issues with identifying novel attacks and handling memories associated with a huge dataset. The significance of high IDS performance was discussed in [33]. Moreover, it presented the different advantages and disadvantages related to the different techniques used to accelerate IDS performance.

Altwaijry et al. [34] suggested improving the accuracy of R2L attack types by utilizing the Bayesian network. Experiments conducted with different KDD99 data set's feature subset led to better results for R2L attack types and had a detection rate 85.35%. Shrivas et al. [35] proposed using an ensemble of Bayes net and artificial neural network (ANN) to classify attacks and normal data for NSL-KDD data sets. The proposed method gave a value of 98.07% with 35 features in the case of the gain ratio feature selection method. Bhavsar et al. [36] proposed a method to classify different types of attacks by using a support vector machine (SVM) possessing different kernel functions. This proposed SVM method with RBF kernel function resulted in a higher classification accuracy value of 98.57%, along with a 10-fold cross validation for the NSL-KDD data set. Dhanabal et al. [33] utilized the NSL-KDD data set and implemented it on support vector machine (SVM), J48, and Naïve Bayes so that they can classify attacks and normal samples. Based on the findings, C4.5 gave the best accuracy for all attack types, considering the normal data that possess 6 feature subsets.

In order to address the issues related to the huge data handling required, we take the advantage of Information Gain (IG)'s parallel computing capabilities. Moreover, we trained the the hybrid approach using a multi-core accelerator platform. The 41 features of the NSL-KDD dataset were reduced to fit the best match in terms of accuracy and usability. We also evaluated the required training time for IDS development. Finally, the hybrid model's detection accuracy was tested for various attack type classifications.

3. Confusion matrix

As seen in Table 1, a confusion matrix is used to represent the information related to the actual and predicted classifications performed by the classification system.

Where,

a = number of correct predictions when an instance is considered negative

b = number of incorrect predictions when an instance is considered positive

c = number of incorrect of predictions when an instance is considered negative

d = number of correct predictions when an instance is considered positive

The accuracy (AC) = total number of correct predictions.

$$AC = \frac{a+d}{a+b+c+d}$$

The true positive rate (TP) = correctly identified positive cases

$$TP = \frac{d}{c+d}$$

The false positive rate (FP) = negative cases that have been incorrectly classified as positive

$$FP = \frac{b}{a+b}$$

The true negative rate (TN) = negative cases that were correctly classified

$$TN = \frac{a}{a+b}$$

The false negative rate (FN) = positive cases that have been classified incorrectly as negative

$$FN = \frac{c}{c+d}$$

4. Classification techniques

Classification is a type of data mining method and is just one of the many classification algorithms currently in use. It works in a manner that may be similar to other techniques, such as decision trees and neural networks. To make its prediction, these techniques use several ways to analyse the available data [33].

- Decision tree: This technique involves the division of the classification problem into several sub-problems. It involves the creation of a decision tree, which can then be utilized to come up with a model that can be applied for the purpose of classification.
- Neural networks: This refers to a set of statistical learning models driven by biological neural networks. These networks are utilized to approximate or estimate functions that normally rely on a large amount of training data.
- Nearest neighbour: In this method, all supplied classes are saved through training data set and new classes are classified based on a similarity measure. Moreover, all the discussed methods are known for their inherent drawbacks and salient features. It takes time to build a decision tree. Thus, when data set size increases, the nearest neighbour method becomes significantly more time consuming. Neural network functions best if numerical data is used; this requires the transformation of the textual data found in the data set into a numerical value.

Because of the aforementioned drawbacks, the idea of utilizing a hybridised approach that involves some optimisation technique was initiated. Hybridization must take into account the existing algorithm that could function well with the available data set and the problem domain.

5. Functionality overview of proposed model

The following steps are involved in developing an effective intrusion detection hybrid model that has higher accuracy and performance:

1. Choosing a proper dataset that has quality data such as NSL KDD. Further details about

NSL KDD dataset is found in Section 5.1.

2. Apportioning the dataset into 20% test and 80% train for the purpose of the experiment. Further detail is found in Section 5.2.

3. The pre-processing phase. This phase allows the reduction or elimination of the noise forced on the data. This is done in order to try and store the significant information only. On the other hand, there is an attempt at simplification of the subsequent treatments by making use of some of the most commonly used techniques such as correction and normalization. Further detail is found in Section 5.2.

4. Building the hybrid model consisting of the following classifiers such as J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and NaiveBayes.

   There are two steps involved in the classification phase: supervised / unsupervised learning and the recognition and decision step. The latter is often used to increase the recognition rate and improve the system performance. Some of the steps needed to create a classification system include:

   - Number representation involves dataset pre-processing from a training set by representing the dataset and choosing the important features.
   - Training classifier is considered the learning step where the model or classifier is built.
   - Classification through the use of a test data that can be used to estimate the classification rules' accuracy. Later, if the accuracy is deemed acceptable, the classification rules can then be used on the new data tuples.

5. Using best classifier to choose the features by using VOTE scheme and Information Gain (IG). The phase of features extraction is the process of determining which parameters can be utilized to provide an accurate character representation to the machine. Some examples taken from families of current primitives include statistical features and structural features. The phase of features extraction has the capacity to improve the accuracy of the classification performance by only selecting the important terms and getting rid of the noisy terms.

6. Developing a model that exhibits the best performance and accuracy. Further detail can be found in Section 5.3.

7. The post processing phase implements correction methods to give a better recognition rate.

5.1 NSL-KDD dataset description

NSL-KDD dataset represents the KDDcup99 dataset's refined version [33]. The NSL-KDD dataset is made up of a large amount of data. Thus, the NSL-KDD data set that was under consideration for training is equivalent to 10% of the main data set. This equates to 494,020 connection vectors and they are labelled as either attack or normal. Many researchers performed various analyses on NSL-KDD dataset and implemented different tools and techniques. Nonetheless, their common aim was to come up with effective IDS. A detailed NSL-KDD dataset analysis using different machine learning techniques was performed with the use of a WEKA tool and discussed in [37].

It is a challenging task to handle huge data as in the NSL-KDD dataset and accelerate IDS performance. Because affordable multi-core hardware platforms are now available, the significance of accelerating the IDS' data handling capabilities has started to attract more interest. Convolutional neural networks that are hardware accelerated and which can be used in image processing applications were developed in [38]. Farabet et al. [39] assessed the performance of FPGA, software, and ASIC implementations and their evaluation revealed a speedup in terms of custom hardware implementation. Microsoft [40] came up with an FPGA-based specialised hardware that aimed to accelerate deep convolutional neural networks so that they can be applied in data centres. They observed very high energy efficiency and significant performance improvement when using TFLOPS. Potluri et al. [41] used GPU-based acceleration for DNN training to classify images and recognise characters. These research studies showed that the time needed for training was significantly reduced by DNN's parallel computing capabilities in training.

There were three major refinements performed on the KDD dataset:

1. Removal of redundant records to allow the classifier to come up with an unbiased result.
2. An adequate number of records are made available in the test and train datasets. These records are reasonably rational and it allows for the execution of experiments on the complete set.
3. From each difficult level group, the amount of selected records is inversely proportional to the record percentages from the original KDD dataset.

In this paper, we have used the NSL KDD dataset for the reasons above. Each record has 41 attributes representing different flow features. Each sample is labelled either normal or attack type. The attribute details, namely the attribute name, sample data, and attribute description are shown in [33]. The NSL-KDD dataset's features have different data types. Table 2 shows the various data types and feature numbers. Aside from normal data, records that correspond to the 39 different attack types are found in the NSL-KDD dataset. All of these attack types can be categorised into four attack classes.

The attacks that were replicated in our experiments can be classified into one of the four types [33??] presented below:

1. Denial of service attack (DOS): This attack type happens when an attacker prevents valid users from accessing the network by consuming the memory or the computer's resources. This makes the system incapable of handling valid requests. There are several examples of DOS attacks: 'neptune,' 'teardrop,' 'ping of death (pod),' 'back', 'mail bomb', 'smurf' and 'land'.

2. Users-to-root attack (U2R): This attack type occurs when an attacker gains access to the system via a valid user account. It is able to gain access to the systems root component by exploiting existing system weaknesses. Some types of U2R attacks include 'buffer overflow', 'load-module', 'rootkit', and 'perl'.

3. Remote-to-local attack (R2L): This attack type happens when an attacker who does not own an account uses existing machine vulnerabilities to locally access a legitimate user account. Some of the R2L attacks types include 'phf', 'warezclient', 'warezmaster', 'spy', 'ftp write', 'imap', 'multihop', and 'guess passwd'.

4. Probing attack (PROBE): This attack type happens when an attacker dodges the security and obtains data from the computers in the network. Some of the PROBE attacks include 'nmap', 'ipsweep', 'satan', and 'portsweep'.

Therefore, we have considered five classes: Normal Class, DoS Class, Probe Class, R2L Class and U2R Class. In more details, denial of service (DoS) has 10 attack types, probing (Probe) has 6 attack types, unauthorised access from a remote machine (R2L) has 16 attack types, and unauthorized access to local super user (U2R) has 7 attack types. Table 3 provides an overview on the NSL-KDD datasets that were used in this study for the testing and training of the developed IDS. This table shows the percentage of the particular records and the number of data elements in the entire dataset.

Note that, the NSL-KDD data took into account the following protocols: UDP, TCP, and ICMP.

5.2 Data capture and feature selection

In this paper, the detectors were trained using the NSL-KDDTrain+20%. The NSL-KDDTrain+20% is made up of 25192 instances, 13449 of which are normal data and 11743 are considered attack data. Two operations were performed on the NSL-KDD before the features were selected: data set pre-processing and normalization.

A. The proposed pre-processing phase

The decision trees classification only utilizes numerical values for the processes of training and testing. Thus, to turn the non-numerical values into numerical values, a pre-processing phase is required. In the proposed model, pre-processing involves the following main tasks:

1) Conversion of the non-numerical dataset features to numerical values: Features 2, 3 and 4 or the protocol type, service and flag were all considered non-numerical. Specific values were assigned to each variable to convert these features in the test and train data set to numerical types (e.g. TCP = 1, UDP = 2 and ICMP = 3).

2) Transform the attack types into its numeric categories at the end of the dataset: 1 is assigned as the normal data. 2, 3, 4 and 5 are used to represent attack types of DoS, Probe, R2L and U2R, respectively.

3) Preparing dataset: The NSS-KDD dataset is used because this dataset is valuable in the system, however it needs some pre-processing. In this paper, the Information Gain (IG) detector is based on the Mutual Information (MI), where the MI process works as follows: (i) Only one packet is inserted into the system in this phase so that the term frequency (TF) can be computed for each token. TF represents the total number of single tokens given a specific packet. Therefore, each token's percentage is computed for this packet. (ii) The next step involves the calculation of the mutual information (MI) for every token as shown by Equation 1. The mutual information for two random variables is in fact an amount that measures and represents the mutual dependence for these two random variables. The bit is the most common measurement unit for mutual information. (ii) Afterwards, the top N MI value are selected to create the vector for this given packet.

$$MI(X;C) = \sum_{i=0}^{n} P(X = x, C = c).log\left(\frac{P(X = x, C = c)}{P(X = x)P(C = c)}\right) \qquad (1)$$

Where, *MI* corresponds to mutual information, *C* represents class, which can either be normal or anomalous, *X* corresponds to the set of *x* vectors, *P(C)* corresponds to the probability of class records being normal or anomalous, *P(X)* corresponds to the probability of a token being classified as either intrusion or normal, and *P(X,C)* represents the probability of a token appearing in the specific class. Based on the theorem of total probability and Bayes theorem, using the vector $\vec{x} = x_1, x_2, \ldots, x_n$ for document *d*, the probability of *d* belonging to category *c* is represented by the following (See Equation 2):

$$P(C = c|\vec{X} = \vec{x}) = \frac{P(C = c).P(\vec{X} = \vec{x}|C = c)}{\sum_{i=0}^{n} P(C = K).P(\vec{X} = \vec{x}|C = K)} \quad (2)$$

Where, *K* represents the class as either being intrusion or benign, $\vec{x}$ corresponds to the set of *x* vectors, and the term frequency value.

Each vector corresponds to one packet. Moreover, each vector has *N* tokens. Every token also possesses its own private and specific index. This just means that if the token is found on a packet, the token's TF value will be placed on its specific index. On the other hand, if the token is not found in this packet, the TF value will then be zero. Moreover, the TF value will still be put on a specific index found in the vector. A label that corresponds to the value for each packet, which in turn corresponds to the type of the packet, is found at the end of the vector. However, the packet is considered an intrusion if this value is equal to 1. If the value is 0, then the packet is considered a normal (benign).

B. Normalization

Because the NSL-KDD dataset features can either have continuous or discrete values, they will have different ranges for the features value, thus making them incomparable. Therefore, min-max normalization was used to normalize the features. This also allowed for the mapping of all the various values for every feature in the [0, 1] range.

To ensure a small table size for the detector, Information Gain was used to reduce the 41 features to 8. Only features with IG over 0.40 were chosen, that is: 5,3,6,4,30,29,33 and 34. Table 4 demonstrates the mapping for the IG selected features, it also shows the name and a short description.

Generally, it is desired to have very low false alarm and very high detection rates. However, a trade-off exists between these two measures. It is recommended to have a high number of GA generation run in order to increase the DR. Moreover, for FA reduction, one should reduce the detection radius from the 0.4 utilized in this experiment to 0.1. However, doing so will lead to a rise in the amount of detectors required to fill up the search space.

5.3 The proposed model

The proposed model's Pseudo code is presented by Algorithm 1 below

***Algorithm 1** Proposed Model*

procedure *model()*
InputFn= *NSL-KDD data set possessing 41 features f1,f2,f3.....f42*
Reduce *41 features to 8 features based on a number of the proposed filters*
Use Vote *scheme*
Develop *a robust model M*
Propose *the model*
  for every *feature* F_n
    Provide F_n *to J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and Naïve Bayes using NSL-KDDTrain+20%*
    Calculate
      A1= J48 model accuracy
      A2= Meta Paging model accuracy
      A3= RandomTree model accuracy
      A4= REPTree model accuracy
      A5= AdaBoostM1 model accuracy
      A6= DecisionStump model accuracy
      A7= NaiveBaye model accuracy
      E= Ensemble representing J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and NaiveBayes with NSL-KDDTrain+20%
    Compare *of the accuracy of A1, A2, A3, A4, A5, A6, A7, E*
Select the best model M= E

6. Experimental results and analysis

This section will present the experiment setup and the analysis of results. Subsection 6.1 explains the experimental setup and vote model while Subsection 6.2 presents the results and analysis.

6.1 Experiment setup

Several standard data mining processes like clustering, data cleaning and pre-processing, classification, visualisation, regression, and feature selection have already been implemented in WEKA. WEKA is an automated data mining tool that is utilized to conduct our classification experiments for the 20% NSL-KDD dataset. The data set is made up of different classes of attacks such as DoS, U2R, R2L, and Probe.

As mentioned in Section 5, the dataset that needs to be classified is pre-processed and normalized so that it has a range of {0-1}. The dataset is classified using different classifiers. Then, the classifier with the best accuracy is applied for feature selection. Two approaches are used for feature selection: search method and subset attribute evaluator. Finally, the model is created by using the selected attributes and the best classifier as shown in the following WEKA details:

Scheme: Vote

Options: -S 1 -B "weka.classifiers.trees.J48 -C 0.25 -M 2" -B "weka.classifiers.trees.RandomTree -K 6 -M 1.0 -V 0.001 -S 1" -B "weka.classifiers.meta.Bagging -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.REPTree -- -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0" -B "weka.classifiers.meta.AdaBoostM1 -P 100 -S 1 -I 10 -W weka.classifiers.trees.DecisionStump" -B "weka.classifiers.meta.Bagging -P 100 -S 1 -num-slots 1 -I 10 -W weka.classifiers.trees.REPTree -- -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0" -B "weka.classifiers.bayes.NaiveBayes " -R AVG

Relation: KDDTrain-20Percent-revised

6.2 Result analysis

In this method, classification of 80% training data was done through j48, random tree, Naïve Bayes, and the proposed model. Comparisons regarding the true positive (TP) rate, accuracy, and false positive (FP) were performed. The classifier with the highest accuracy was considered the best.

Table 5 shows that the proposed hybrid model exhibited the highest percentage (99.81) in terms of successfully classifying the instances. Moreover, it exhibited the lowest false positive rate (0.003) and highest percentage for true positive (TP) rate (0.997). Thus, among the four classifiers, the proposed model is proven to be superior.

Wrapper method

Wrapper method: A subset evaluator is utilized in the wrapper method to generate all possible subsets from a feature vector. Afterwards, a classification algorithm is implemented to induce a classifier from the subset's features. The subset of features with the best performing classification algorithm will be considered. For example, if there are 10 features, the evaluator will try to look for the subset that has those 10 features: 1st attribute: 3 features, 2nd attribute: 3 features, and 3rd attribute: 4 features. The classifier is applied to all the subsets and the subset that gives the best accuracy is determined. Search techniques like random search are used by the evaluator to find a subset.

Feature reduction and feature selection through best classifier machine learning and data mining were utilized to improve the classifier's accuracy. A dataset is made up of a vast amount of features, but not all those features are vital. Feature selection and reduction of unwanted features are some of the most important factors that influence the increase in the classifier's efficiency. There are two techniques that can be used for feature selection and reduction: filter method and wrapper method.

ROC curve

The receiver operating characteristic (ROC) curve is a tool for visualisation that can be utilized to determine whether a classifier is appropriate in terms of cost sensitivity. ROC can analyse the performance through these four basic classification types:

False Positive (FP) – incorrect positive prediction, True Positive (TP) – correct positive prediction, False Negative (FN) – incorrect negative prediction, and True Negative (TN) – correct negative prediction. In Figure 2, the curve's x-axis represents the false positive while the y axis represents the false negative. The area found under curve with the value of (0.999) indicates that it is an appropriate classifier.

Table 7 shows that the proposed hybrid model exhibited the highest percentages in all classes (99.7, 99.9, 96.2, 99.1, and 97.9) in terms of successfully classifying the instances. Thus, among the four classifiers, the proposed model is seen as the best classifier.

7. CONCLUSION AND FUTURE WORK

Results from the analysis of the NSL-KDD dataset revealed that it is the top candidate data set that can be used to test and simulate IDS performance. The proposed hybrid model for dimensionality reduction improves the accuracy rate and reduces the detection time. The analysis performed on the NSL-KDD dataset through the help of tables and figures has allowed the researcher to gain a clearer dataset understanding. It also shows that majority of attacks are done using the TCP protocol’s inherent drawbacks. May be summarize final performance numbers and accuracy here.

For future studies, it is recommended that researchers should study the possibility of applying optimising techniques to come up with an intrusion detection model that has a better accuracy rate. We will further expand on this area in our future work through the implementation of a fully distributed Network IDS. Moreover, it will apply other techniques to ease intercommunication among NIDS.

Acknowledgement

This research was supported, in part, by Zayed University Research Office, Research Incentive Grant, R15121.

References

[1] Eduardo DelaHoz, EmiroDeLaHoz Andre´sOrtiz, JulioOrtega and BeatrizPrieto, PCA filtering and probabilistic SOM for network intrusion detection, in: Special Issue: Advances in Computational Intelligence in Elsevier—Neurocomputing, vol. 164, 2015, pp. 71–81.

[2] Ujwala Ravale, Nilesh Marathe, Puja Padiya, Feature selection based hybrid anomaly intrusion detection system using K means and RBF kernel function, in: Proceeding of International Conference on Advanced Computing Technologies and Applications, ICACTA-2015, in: Procedia Computer Science, vol. 45, Elsevier, 2015, pp. 428–435.

[3] D.P. Gaikward, Ravindra c Thool, Intrusion detection system using bagging with partial decision tree base classifier, in: Proceeding of International Conference on Advanced in Computing, Communication and Control, ICAC3’15, in: Procedia Computer Science, vol. 49, Elsevier, 2015, pp. 92–98.

[4] Sunil Nilkanth Pawar1, Rajankumar Sadashivrao Bichkar, Genetic algorithm with variable length chromosomes for network intrusion detection, Int. J. Autom. Comput. 12 (3) (2015) 337–342.

[5] D. E. Denning, “An Intruison-Detection Model,” in IEEE Symposium on Security and Privacy, 1986, 1986, pp. 118–131.

[6] D. Anderson, T. Frivold, and A. Valdes, “Next- generation Intrusion Detection Expert System (NIDES): A summary,” SRI Int., no. May 1995, p. 47,1995.

[7] M. Lincoln Laboratory, “DARPA Intrusion Detection Data Sets.” [Online]. Available: https://www.ll.mit.edu/ideval/data/. [Accessed: 07- Apr-2016].

[8] J. McHugh, “Testing Intrusion detection systems: a [33] critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory,” ACM Trans. Inf. Syst. Secur., vol. 3, no. 4, [34] pp. 262–294, 2000.

[9] Fangjun Kuang, Siyang Zhang, Zhong Jin, Weihong Xu, A novel SVM by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection, Soft Computing 19 (2015) 1187–1199.

[10] M. Aldwairi, Y. Khamayseh, and M. Al-Masri, Application of artificial bee colony for intrusion detection systems, *Sec. and Commun. Netw*, vol. 8, no. 16, November 2015, pp. 2730-2740. DOI=http://dx.doi.org/10.1002/sec.588

[10] Iftikhar Ahmad, Muhammad Hussain, Abdullah Alghamdi, Abdulhameed Alelaiwi, Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components, Neural Comput. 24 (2014) 1671–1682.

[11] Chun Guo, Yajian Zhou, Yuan Ping, Zhongkun Zhang, Guole Liu, Yixian Yang, A distance sum-based hybrid method for intrusion detection, in: Appl. Intell., vol. 40, Springer, 2014, pp. 178–188.

[12] Saurabh Mukherjee, Neelam Sharma, Intrusion detection using naive bayes classifier with feature reduction, in: Proceedings in 2nd International Conference on Computer, Communication, Control and Information Technology, C3IT-2012, in: Procedia Technology, vol. 4, Elsevier, 2012, pp. 119–128.

[13] S. Snapp, J. Brentano, and G. Dias et al., “DIDS (Distributed Intrusion Detection System) – motivation, architecture, and an early prototype,” In Proceedings of the 14th National Computer Security Conference, October 1991.

[14] M. Crosbie and G. Spafford, “Defending a computer system using autonomous agents,” Technical Report 95-022, COAST Laboratory - Purdue University, 1994.

[15] F. Hosseinpour, A. Meulenberg, S. Ramadass, P. Vahdani, and Z. Moghaddasi, “Distributed agent based model for intrusion detection system based on artificial immune system,” Int. J. Digit. Content Technol. its Appl., vol. 7, pp. 206–214, 2013.

[16] N. Afzali and R. Azmi, “MAIS-IDS: a distributed intrusion detection system using multi-agent AIS approach,” Eng. Appl. Artif. Intell. vol. 35, pp. 286–298, 2014.

[17] J. Singh and M. J. Nene, “A Survey on Machine Learning Techniques for Intrusion Detection Systems,” [35] Int. J. Adv. Res. Comput. Commun. Eng., vol. 2, no. 11, pp. 4349–4355, 2013.

[18] S. K. Wagh, “Survey on Intrusion Detection System using Machine Learning Techniques,” Int. J. Comput. Appl., vol. 78, no. 16, pp. 30–37, 2013.

[19] C. Qiu, J. Shan, B. Polytechnic, and B. Shandong, “Research on Intrusion Detection Algorithm Based on BP Neural Network,” vol. 9, no. 4, pp. 247–258, 2015.

[20] L. Vokorokos, A. Baláž, and M. Chovanec, “Intrusion detection system using self organizing map,” Informatica, vol. 6, no. 1, pp. 1–6, 2006.

[21] J.-P. Planquart, "Application of Neural Networks to Intrusion Detection," 2001.

[22] S. K. Sahu, S. Sarangi, and S. K. Jena, "A detail analysis on intrusion detection datasets," Souvenir 2014 IEEE Int. Adv. Comput. Conf. IACC 2014, pp. 1348– 1353, 2014.

[23] J. Allen, C. Alan, F. William, M. John, P. Jed, and S. Ed, "State of the Practice of Intrusion Detection Technologies, Technical Report no. CMU/SEI-99-TR- 028," Tech. Rep., no. January, p. 221, 2000.

[24] V. Paxson, "Bro: a system for detecting network intruders in real-time," Comput. Networks, vol. 31, no. 23–24, pp. 2435–2463, 1999.

[25] R. Sekar, Y. Guang, S. Verma, and T. Shanbhag, "A high-performance network intrusion detection system,"Proc. 6th ACM Conf. Comput. Commun. Secur. - CCS '99, pp. 8–17, 1999.

[26] H. Lu, K. Zheng, B. Liu, X. Zhang, and Y. Liu, "A memory-efficient parallel string matching architecture for high-speed intrusion detection," IEEE J. Sel. Areas Commun., vol. 24, no. 10, pp. 1793–1803, 2006.

[27] A. Das, D. Nguyen, J. Zambreno, G. Memik, and A. Choudhary, "An FPGA-based network intrusion detection architecture," IEEE Trans. Inf. Forensics Secur., vol. 3, no. 1, pp. 118–132, 2008.

[28] N. S. Artan, R. Ghosh, Y. Guo, and H. J. Chao, "A 10- Gbps High-Speed Single-Chip Network Intrusion Detection and Prevention System," IEEE GLOBECOM 2007-2007 IEEE Glob. Telecommun. Conf., pp. 343– 348, 2007.

[29] G. Vasiliadis, S. Antonatos, M. Polychronakis, E. Markatos, and S. Ioannidis, "Gnort: High performance network intrusion detection using graphics processors," Recent Adv. Intrusion Detect., pp. 116–134, 2008.

[30] S. Bastke, "Combining statistical network data, probabilistic neural networks and the computational power of GPUs for anomaly detection in computer networks," Work. Intell. Secur. (SecArt 2009), no. iii, pp. 1–6, 2009.

[31] Q. OuYang, "Theoretical and Mathematical Foundations of Computer Science," Commun. Comput. Inf. Sci., vol. 164, no. January, pp. 154–160, 2011.

[32] Sasanka Potluri and Christian Diedrich, "High Performance Intrusion Detection and Prevention Systems: A Survey," unpublished.

[33]. L. Dhanabal and Dr. S.P. Shantharajah, A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms, International Journal of Advanced Research in Computer and Communication Engineering, 4, 2015, 446-452.

[34] H. Altwaijry, and S. Algarny, Bayesian based Intrusion Detection System, Journal of King Saud University Computer and Information Sciences. 24,2012, 1-6.

[35] A. K. Shrivas and A. K. Dewangan, An Ensemble Model for Classification of Attacks with Feature Selection based on KDD99 and NSL-KDD Data Set, International Journal of Computer Applications,99,2014,8-13.

[36] Y. B. Bhavsar and K. C.Waghmare, Intrusion Detection System Using Data Mining Technique: Support Vector Machine, International Journal of Emerging Technology and Advanced Engineering, 3, 2013, 581-586.

[37] D. a. M. S. Revathi, “A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection,” Int. J. Eng. Res. Technol., vol. 2, no. 12, pp. 1848–1853, 2013.

[38] D. Cireşan, U. Meier, and J. Schmidhuber, “Multi-column Deep Neural Networks for Image Classification,” Int. Conf. Pattern Recognit., no. February, pp. 3642–3649, 2012.

[39] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello, “Hardware Accelerated Convolutional Neural Networks for Synthetic Vision Systems,” Proc. Int. Symp. Circuits Syst., 2010.

[40] K. Ovtcharov, O. Ruwase, J. Kim, J. Fowers, K. Strauss, and E. S. Chung, “Accelerating Deep Convolutional Neural Networks Using Specialized Hardware,” pp. 3–6, 2015.

[41] Potluri, Sasanka, and Christian Diedrich. "Accelerated deep neural networks for enhanced Intrusion Detection System." In Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on, pp. 1-8. IEEE, 2016.

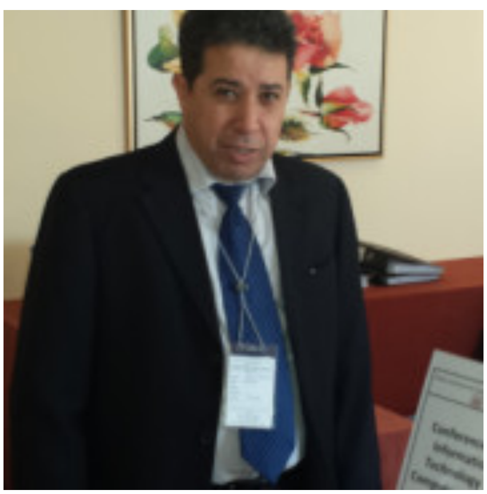
[42] S. A. Aljawarneh, R. A. Moftah, and A. M. Maatuk, “Investigations of automatic methods for detecting the polymorphic worms signatures,” Futur. Gener. Comput. Syst., vol. 60, pp. 67–77, 2016.

Shadi Aljawarneh is an associate professor, Software Engineering, at the Jordan University of Science and Technology, Jordan. He holds a BSc degree in Computer Science from Jordan Yarmouk University, a MSc degree in Information Technology from Western Sydney University and a PhD in Software Engineering from Northumbria University-England. He worked as an associate professor in faculty of IT in Isra University, Jordan since 2008. His research is centered in software engineering, web and network security, e-learning, bioinformatics, Cloud Computing and ICT fields. Aljawarneh has presented at and been on the organizing committees for a number of international conferences and is a board member of the International Community for ACM, Jordan ACM Chapter, ACS, and IEEE. A number of his papers have been selected as "Best Papers" in conferences and journals.

Monther Aldwairi is an associate professor at the College of Technological Innovation at Zayed University since the fall of 2014. He received his B.S. in electrical engineering from Jordan University of Science and University (JUST) in 1998, and his M.S. and PhD in computer engineering from North Carolina State University (NCSU), Raleigh, NC, in 2001 and 2006, respectively. Prior to joining ZU, he was an Assistant and then Associate Professor of Computer Engineering at Jordan University of Science and Technology. He served as the Vice Dean of the Faculty of Computer and Information Technology from 2010 to 2012 and was the Assistant Dean for Student Affairs in 2009. In addition, he was an Adjunct Professor at New York Institute of Technology (NYiT) from 2009 to 2012. He worked at NCSU as Post-Doctoral Research Associate in 2007 and as a research assistant from 2001 to 2006. He worked as a system integration engineer for ARAMEX from 1998 to 2000. Dr. Aldwairi’s research interests are in information, network and web security, intrusion detection, digital forensics, cloud computing, reconfigurable architectures, artificial intelligence and pattern matching.

Dr. Muneer Masadeh Bani Yassein received his B.Sc. degree in Computing Science and Mathematics from Yarmouk University, Jordan in 1985 and M. Sc. in Computer Science, from Al Al-bayt, University, Jordan in 2001. And PhD degrees in Computer Science from the University of Glasgow, U.K., in 2007, He is currently an associate professor in the Department of Computer science at Jordan University of Science and Technology (JUST), Muneer served as Chairman of the department of Computer science from 2008 to 2010,

as Vice Dean of the Faculty of Computer and Information Technology from 2010 to 2012, and from2013-2014.Muneer is currently conducting research in Mobile Ad hoc Networks, Wireless sensors Networks, Cloud Computing, simulation and modelling, development/analysis of the performance probabilistic flooding behaviours in MANET, optimizations and the refinement of service discovery and routing algorithms for mobile device communications in heterogeneous network environments. Bani Yassein has published over 90 technical papers in well reputed international journals and conferences. During his career, he has supervised more than 50 graduate and undergraduate students. Dr. Bani Yassein is member of IEEE and he is a member of the technical programs of several journals and conferences.



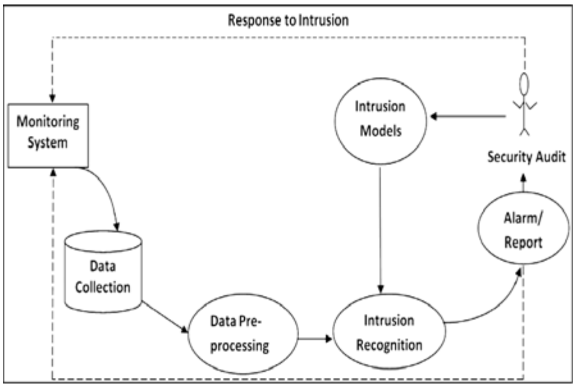


Figure 1 Overall structure of intrusion detection system [1]

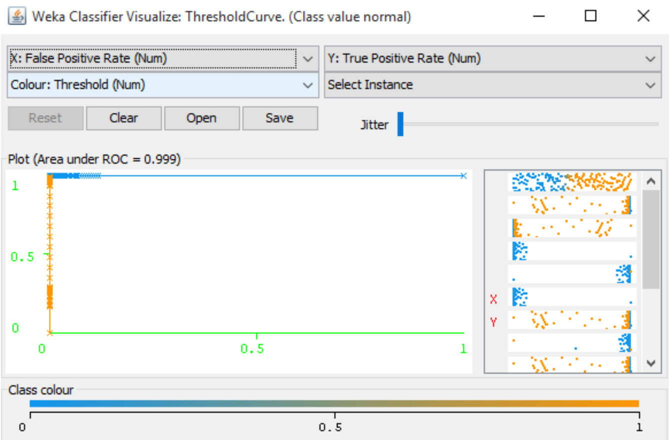


Figure 2: ROC curve for NORMAL class

Table 1. Confusion matrix

		Predicted	
		Negative	Positive
Actual	Negative	a	b
	Positive	c	d

TABLE 2: Features with different data types IN NSL-KDD

Feature Type	Features
Nominal	2, 3, 4
Binary	7, 12, 14, 15, 21, 22
Numeric	1,5,6,8,9,10,11,13,16,17,18,19,20,23,24,25,26,27,28,29,30,31,32,33, 34,35, 36,37,38,39,40,41

TABLE 3: Overview on NSL-KDD data

Data set type	No of data samples					
	Records	Normal	DoS	Probe	U2R	R2L
NSL-KDD Train	125973	67343	45927	11656	52	995
	%	53.46	36.45	9.25	0.04	0.79
NSL-KDD Test	22543	9711	7458	2421	200	2754
	%	43.08	33.08	10.74	0.89	12.22

Table 4: Description of selected features

Feature	Description
5	(src_bytes): Number of data bytes transferred from source to destination in single connection
3	(service): Destination network service used
6	(dst_bytes): Number of data bytes transferred from destination to source in single connection
4	(flag): Status of the connection – Normal or Error
30	(diff_srv_rate): The % of connections that were to different services, among the connections aggregated in count
29	(same_srv_rate): The % of connections that were to the same service, among the connections aggregated in count
33	(dst_host_srv_count): The % of connections that were to the same service, among the connections aggregated in dst_host_count
34	(dst_host_same_srv_rate): The % of connections that were to different services, among the connections aggregated in dst_host_count

Table 5: Comparison of four classifiers

Classifier	TP	FP	Correctly classified instance	Incorrectly classified instance
Naïve Bayes	0.903	0.102	90.2876	9.7124
J48	0.997	0.003	99.74	0.26
Random Tree	0.997	0.003	99.747	0.253
Proposed Model	0.997	0.003	99.81	0.25

TABLE 7: Accuracy in detection of normal and attack network flows by using the J48, SVM, Naïve Bayes and the proposed model classifiers

Classification Algorithm	Class Name	Test Accuracy
J48	Normal	99.8
	DoS	99.1
	Probe	98.9
	U2R	98.7
	R2L	97.9
SVM	Normal	98.8
	DoS	98.7
	Probe	91.4
	U2R	94.6
	R2L	92.5
Naïve Bayes	Normal	74.9
	DoS	75.2
	Probe	74.1
	U2R	72.3
	R2L	70.1
Proposed Hybrid Model	Normal	99.7
	DoS	99.9
	Probe	96.2
	U2R	99.1
	R2L	97.9