A SURVEY ON DIFFERENT MACHINE LEARNING ALGORITHMS AND WEAK CLASSIFIERS BASED ON KDD AND NSL-KDD DATASETS

Rama Devi Ravipati[1]and Munther Abualkibash[2]

[1]Graduate student in the Department of Computer Science, Eastern Michigan University, Ypsilanti, Michigan
[2]School of Information Security and Applied Computing, Eastern Michigan University, Ypsilanti, Michigan

ABSTRACT

Network intrusion detection often finds a difficulty in creating classifiers that could handle unequal distributed attack categories. Generally, attacks such as Remote to Local (R2L) and User to Root (U2R) attacks are very rare attacks and even in KDD dataset, these attacks are only 2% of overall datasets. So, these result in model not able to efficiently learn the characteristics of rare categories and this will result in poor detection rates of rare attack categories like R2L and U2R attacks. We even compared the accuracy of KDD and NSL-KDD datasets using different classifiers in WEKA.

KEYWORDS

KDD, NSL-KDD, WEKA, AdaBoost, KNN, Detection rate, False alarm rate

1. INTRODUCTION

On a computing platform Intrusion is a set of actions that attempts to compromise the integrity, confidentiality, or availability of any resource. An Intrusion Detection System (IDS) is a combination of hardware and software that detect intrusions in the network. It is used to track, identify and detect the intruders. IDS is a combination of either hardware or software or both which is used to detect intrusions in the network. IDS is used to detect unauthorized intrusions that occur in computer systems and networks. The objectives are Confidentiality, Integrity, Availability and Accountability.

Intrusion Prevention System is classified into four types:

a. Network based intrusion prevention system – It analyses the protocol activity by monitoring the entire network for suspicious traffic.
b. Wireless Intrusion prevention system – It analyses the wireless networking protocol by monitoring the entire network for suspicious traffic.
c. Network behaviour analysis – Examines the network to identify the threats such as distributed denial of service.
d. Host based Intrusion Prevention System – An installed software which monitors a host for a suspicious activity by analysing the host.

The most used intrusion detection techniques are signature based and anomaly based.

DOI : 10.5121/ijaia.2019.10301

a. Signature-based IDS refers to the attacks in the network traffic by looking for specific patterns, such as byte sequences [1]. It only detects the known attacks but not new attacks because no pattern is available for those new attacks [2].
b. Anomaly-based IDS is used to detect the unknown attacks. It uses machine learning to create a model and then compare it with the new behaviour. This approach helps to detect the new attacks, but it suffers from false positives. If we use efficient feature algorithms, then classification process becomes reliable [3].

The algorithm that needs fully class labelled data is called a supervised learning algorithm, and some algorithms that do not need class labelled data which are called unsupervised learning algorithms. We also have some algorithms which use properties of both supervised and unsupervised algorithms which are called Semi-supervised algorithms [4].Semi-supervised algorithms do not need all records to be class labelled. An Unsupervised learning algorithm finds the hidden pattern in the data, whereas a supervised learning algorithm finds the relationship between data and its class.

2. RELATED WORK

Many researchers have used intrusion detection system based on supervised learning approach, such as neural network (NN)-based approach and support vector machine (SVM)-based approach. Bonifacio et al [5] proposed an NN for distinguishing between intrusions and normal behaviours. Recently Several hybrid IDS have been proposed which deal with the complexity of the intrusion detection problem by combining different machine learning algorithms. By incorporating a Hierarchical Clustering and SVM yield to a hybrid intelligent IDS that was developed by Shi-Jinn Horng et al[6]. The SVM theory was slightly modified in this research to be used with standard network intrusions dataset that contains labels. Cheng Xiang et al [7] designed IDS to detect intrusions by combining the supervised tree classifiers and unsupervised Bayesian clustering.

Supervised learning strategies for intrusion recognition can just distinguish known intrusions yet unsupervised learning strategies recognize the intrusions that have not been recently learned. The unsupervised learning for intrusion detection includes K-means-based approach and self-organizing feature map (SOM)-based approach.

Deep learning is a part of machine learning that is classified based on the layers present in the artificial neural network. This can be categorized into two types supervised and unsupervised learning. This Deep learning methodology deals with several architectures like Deep Neural Networks, Convolution neural networks, Recurrent Neural Networks etc. Based on the depth or number of layers involved in the network generally tends to yield high accuracy when compared to less number of layers [12]. The working of these structures is as follows: Input layer takes the set of features and the output layer yields the specific set of features that are meant to be there in the targeted output. In between these there are several hidden layers, each of which are fed with the output of the previous layer and extracts the precise set of features i.e. dimensionality reduction and are sent to next layer. We use functions for error minimization and activation functions for the neural network to train and work effectively.

Neuro-fuzzy system is a combination of neural networks with fuzzy logic or fuzzy sets helps to reduce the drawbacks in both the systems and would help to yield better productivity. It consists of a fuzzy system which uses some kind of learning algorithm such as neural network algorithm for deciding the parameters or input for the fuzzy system [13]. The main feature of this network is transparency [14]. Neuro—fuzzy differ from normal neural networks in almost all the ways such as activation functions, connection weights and propagation functions. These Neuro-fuzzy system can be consider as 3 layer system, first layer is a input variables and middle layer consists of

fuzzy rules and final layer consists of output variables. This can also represented as 5 layer system in which 2 and 4 are fuzzy system layers [15].

Decision trees are mostly used for classification and prediction. These decision trees consists of nodes internal nodes represent a test and the branch denotes the outcomes of the test and each external or leaf node consists of the class label to which a particular data or the object can be classified. Decision trees are more popular and less computationally expensive when compared to the other machine learning classifiers. These are good in several cases but when it comes to estimation of a value based on the continuous attribute it fails as it couldn't predict the possible outcome value. They need lots of training data for each class label included, if we train It with less number of examples for each class label then it could not make proper classification and is prone to errors and for growth of the decision tree we need to split it to find the best fit which is tedious and involves lots of effort and computation.

The KDD dataset is the most used one for training the algorithm. KDD dataset is a consistent and widely used dataset in the field of network intrusion detection. It is the subset of DARPA-98 dataset. KDD dataset is a multi-variate dataset. The dataset contains 4.8 million instances. The characteristics of the attributes used in the dataset are categorical and integer in nature. It has forty-two attributes. The KDD dataset mainly consists of four types of attacks –

1. DOS (Denial of Service): DOS is a type of attack category in which it consumes the victim's resources so that it can unable to handle legitimate requests – e.g. SYN flooding.
2. R2L (Remote to Local): R2L is a unauthorized access from a remote machine, in which the attacker intrudes enter into a remote machine and gains the local access of the victim machine. E.g. password guessing.
3. U2R (User to Root): U2R is a unauthorized access to the root privileges, in which the attacker uses a normal account to login into a victim system and tries to gain rootprivileges by exploiting some vulnerability in the victim system. E.g. buffer overflow attacks.
4. Probing: Probing attacks main objective is to gain information about the remote victim. E.g. port scanning.

The training dataset consists of 21 different attacks. The known attack types are those present in the training dataset while the novel attacks are the additional attacks in the test dataset i.e. not available in the training datasets. The attack types are categorised into four: DOS, Probe, U2R and R2L as in KDD dataset. Each class consists of a group of attacks.

Table 1. Classification of attack types in KDD Dataset.

Classes	Attack types
Normal	Normal: 97277
DOS	Smurf: 280790 Neptune: 107201 Teardrop: 979 Pod: 264 Land: 21
R2L	Warezclient: 1020 Guess_passwd: 53 Warezmaster: 20 Imap: 12 ftp_write: 8 Multihop: 7 Phf: 4 Spy: 2
U2R	Buffer: 30 Rootkit: 10 Loadmodule: 9 Perl: 3
PROBE	Satan: 1589 Ipssweep: 1247 Portsweep: 1040 Nmap: 231

In the KDD dataset the performance of evaluated systems is highly affected because of two important issueswhich results in very poor evaluation of anomaly detection approaches. To overcome with these issues and increase the performance, they proposed a new dataset called NSL-KDD, which consists of only selected records from the complete KDD dataset.
The benefits of using the NSL-KDD dataset are:

1. No redundant records in the training set, so the classifier will not produce any biased results.
2. There is no duplicate record in the test set.
3. The percentage of records in the original KDD dataset is directly proportional to the number of selected records from each difficult level group.

3. OVERVIEW OF OUR ALGORITHM

According to the characteristics of Intrusion detection problem the framework consists of four modules: feature extraction, data labelling, design of weak classifier, and construction of strong classifiers.

Feature Extraction: According to Sung and Mukkamala [8], Kayacik, Zincir-Heywood et al [9]. and Lee, Shin et al., [10] most published results observing feature reduction on the KDD dataset is trained and tested on the '10%' training set only. The full training dataset contains 4,898,431 records. It contains 22 different attack types. The dataset is unlabelled where as 10% training set

is labelled. Neural networks require floating point numbers for the input neurons, in the range of [−1, 1], and for target neurons floating point numbers should be in the range of [0, 1]. All features were pre-processed to fulfil this requirement. For nominal features with distinct values like User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), Transmission Control Protocol (TCP) we use UDP= [0,1], ICMP= [1,0] and UDP= [-1, -1]. For nominal features with large number of distinct values we assign rank to them according to the number of occurrences in the test set. For the numeric feature's 'duration', 'src bytes' and 'dst bytes', were started with the removal of outliers before scaling the values to the range [−1, 1]. The data may be either continuous or categorical.

Data Labelling: Algorithm uses supervised learning in which sets of data should be labelled for training. +1 is used for normal data and -1 is used for attacked data.

Design of weak classifier: As individual weak classifiers are simple and easy to implement but its classification accuracy is low, the Adaboost algorithm requires a group of weak classifiers. Decision stumps are used in designing of weak classifiers.

Construction of the Strong Classifier using Machine Learning Algorithm: A strong classifier is obtained by combining weak classifiers. Its classification accuracy is high compared to weak classifier. We can construct strong classifiers by using any of the machine learning algorithms.

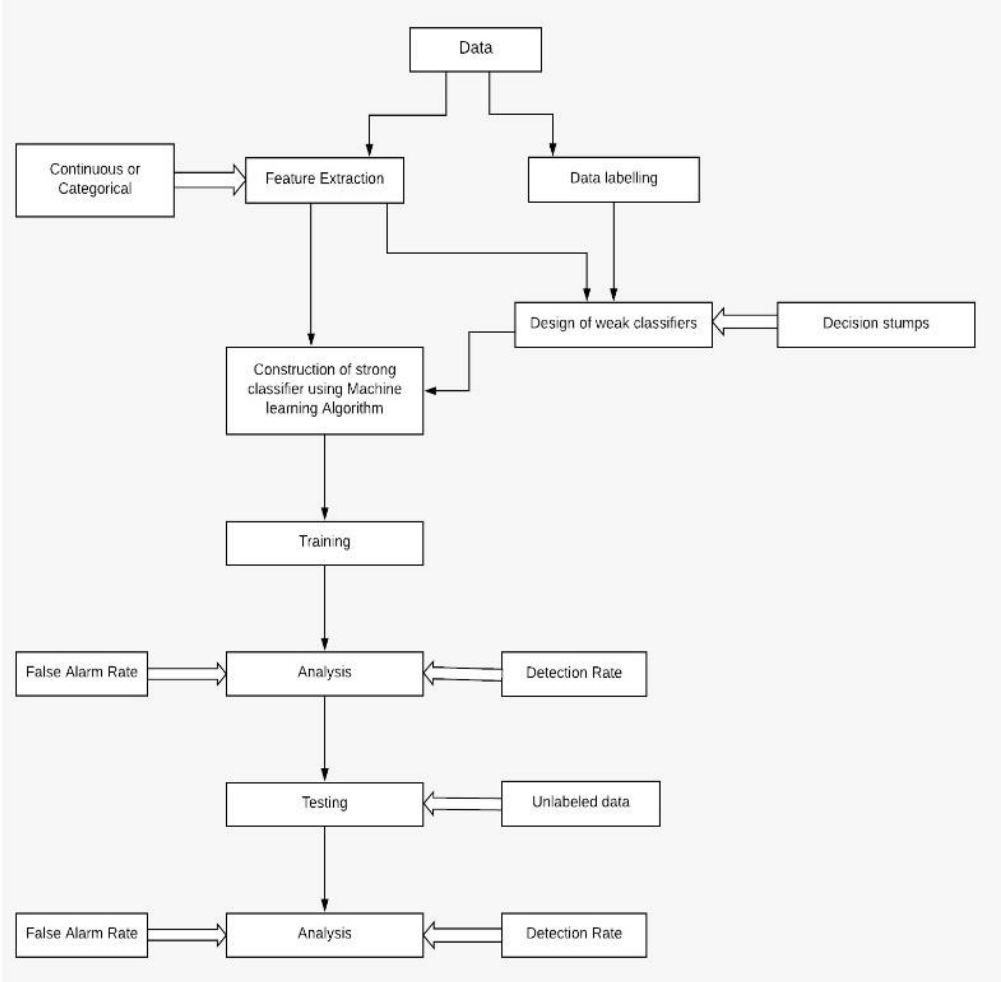

Fig 1. Framework of our Algorithm

Training: After the construction of strong classifiers the data is classified into train and test datasets. We train the data according to our use.

Analysis: On the trained data we perform analysis and finds out the False Alarm rate and the Decision Rate.

Testing: The data which we use is unlabeled data. Now we train the test data according to our use. Next we perform analysis on data and finds the False Alarm Rate and Decision Rate.

Initially started with the implementation of a paper "AdaBoost-Based Algorithm for Network Intrusion Detection."[7] When we implemented the AdaBoost algorithm for KDD dataset we got the same accuracy and false alarm rate as specified in the paper. In this algorithm, we initially extract features and specify data labelling. Later for AdaBoost algorithm we need a group of weak classifiers. In this algorithm, we use decision stump as weak classifiers. Finally, we obtain a strong classifier by combining all the weak classifiers and this will have the higher classification accuracy when compared to weak classifiers.

AdaBoost Algorithm

1. Initialize weights for each data point as $w(x_i, y_i) = \frac{1}{n}$, I = 1,….,n.
2. For iterations m= 1,…..,M

    a. Fit weak classifiers to the data set and select the one with the lowest weighted classification error:

$$€_m = E_w[1_{y \neq f(x)}] \quad (1)$$

    b. Calculate the weight of the m classifier:

$$\theta_m = \frac{1}{2}\ln\left(\frac{1-\epsilon_m}{\epsilon_m}\right) \quad (2)$$

3. Update the weight for each datapoint as:

$$w_{m+1}(x_i, y_i) = \frac{w_m(x_i, y_i)\exp[-\theta_m y_i f_m(x_i)]}{z_m} \quad (3)$$

Where z_m is a normalization factor that ensures the sum of all instance weights is equal to 1.

After M iteration we can get the final prediction by summing up the weighted prediction of each classifier.

Table 2: Number of samples in KDD dataset

Dataset	Normal	Attacks			
		Dos	R2L	U2R	Probe
Training	97278	391458	1126	52	4107
Testing	60593	229853	16189	228	4166

The above table shows the number of attacks and normal data the both training and testing dataset have.

Table 3: Results with uniform initial weights.

Training set		Testing set	
Detection Rate	False Alarm Rate	Detection Rate	False Alarm Rate
99.25%	2.766%	90.738%	3.428%

The above table shows the Detection accuracy and false alarm rate of training and testing datasets obtained after applying Adaboost algorithm. Basically in Adaboost we can update the weights to find the better detection accuracy and false alarm rate.

Next, we applied a Decision tree algorithm for the KDD dataset. A decision tree is one of the widely used supervised machine learning algorithms which performs both regression and classification tasks. The intuition behind the Decision tree algorithm is simple, yet also very powerful. It requires relatively less effort for training the algorithm and can be used to classify non-linearly separable data. It is very fast and efficient compared to KNN and other classification algorithms. Information Gain, Gain Ratio and Gini Index are the most commonly used Attribute Selection Measures. A Decision tree works as follows:

a. First select the best attributesand then split the records.
b. Make the selected attribute as decision node and now break the dataset into smaller subsets.
c. It starts building the tree by repeating this process recursively for each child node until any one of the condition matches:
    1. All the subsets belong to the same attribute value.
    2. There are no more remaining attributes.
    3. There are no more instances.

The Next algorithm we used was Multilayer Perceptron (MLP). An MLP can be viewed as a logistic regression classifier. It consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a activation function. MLP utilizes a supervised learning technique called back propagation for training. Multi-layer perceptron works as follows:

1. For all input neurons i do
2. Set $a_i \leftarrow x_i$ (4)
3. End for
4. For all hidden and output neurons i in topological order do
5. Set $net_i \leftarrow w_{i0} + \Sigma_{j \epsilon pred(i)} w_{ij} a_j$ (5)
6. Set $a_i \leftarrow f_{\log} \ (net_i)$ (6)
7. End for
8. For all output neurons i do
9. assemble a_i in output vector y
10. End for
11. Return y.

Finally, we used a KNN algorithm. KNN is a lazy learning algorithm. In KNN, K is the number of nearest neighbours. The number of neighbours is the core deciding factor.
KNN algorithm works as follows:

1. Load the data.
2. Initialize the value of k.
3. To get the predicted class, iterate from 1 to total number of training data points.
    1. Calculate the distance between test data and each row of training data using Euclidean distance formula.
    2. Sort the calculated distances in ascending order.
    3. Get top k rows from the sorted array.
    4. Get the most frequent class from these rows.
    5. Return the predicted class.

Table 4. Comparison of Detection Accuracy and False Alarm Rate of various algorithms.

Algorithms	**Detection Accuracy**	**False Alarm Rate**
AdaBoost	90.73	3.42
Decision Tree	81.05%	2.85
Multilayer perceptron	80.5%	1.94
KNN	94.17%	5.82
SVM	83.09%	16.90

The Detection Accuracy and False Alarm Rate are calculated for various algorithms by using KDD dataset. The formula to calculate those are given below. They are calculated by using a confusion matrix which is generated after testing the data.

3.1. Parameters used for classification

The confusion matrix consists of four values: True Positive, False Negative, False Positive, and True Negative.

a. True Positive (TP): It is the test result which detects the condition if the condition is present. The TP value in the confusion matrix is at cm [0][0].
b. False Negative (FN): It is the test result which does not detect the condition even if the condition is not present. The FN value in the confusion matrix is at cm [1][0].
c. False Positive (FP): It is the test result which detects the condition when the condition is absent. It is also called a False Alarm rate. The FP value in the confusion matrix is at cm [0][1].
d. True Negative (TN): It is the test result which does not detect the condition even if the condition is present. The TN value in the confusion matrix is at cm [1][1].

Performance can be measured using the

1. Accuracy: The *accuracy* (*AC*) is the proportion of the total number of predictions that were correct. It is given by the relation

$$\text{Accuracy}= \frac{TP+TN}{TP+TN+FP+FN} \quad (7)$$

2. False Alarm Rate: False Alarm rate is calculated as the number of all incorrect predictions divided by the number of true positives. It is given by the relation

$$\text{False Alarm rate} = \frac{FP+FN}{TP} \quad (8)$$

3. Error Rate: Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset. The best error rate is 0.0, whereas the worst is 1.0. It is given by the relation

$$\text{Error rate} = \frac{FP+FN}{TP+TN+FP+FN} \quad (9)$$

4. Recall: Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples.It is also called Sensitivity, True Positive Rate or Detection Rate. It is given by the relation

$$\text{Recall} = \frac{TP}{TP+FN} \quad (10)$$

5. Precision: Precision is defined as the number of true positives over the number of true positives plus the number of false positives.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (11)$$

3.2. WEKA (Waikato Environment For Knowledge Analysis)

In this era of data science where R and Python are ruling the roost, we have another data science tool called Weka.

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.

We have five different classifiers to classify the dataset which are J48, Naive Bayes, AdaBoost, Bagging, and Nearest Neighbor.

1. J48: The J48 classification algorithm works by establishing a decision tree according to values of features. Internal nodes are the attributes whereas leaf nodes are classes. Features that have maximum information gain are used as root nodes which classify the instances more clearly. Then the next attribute with the second highest information gain is used until corresponding class is decided.
2. Naïve Bayes: This classifier is based on probabilistic Bayesian theorem. This technique is capable of performing well with many complicated real-world applications. Naive Bayes classifier assumes features to be independent. It is also suited for high dimensional data.
3. AdaBoost: AdaBoost is used for constructing a strong classifier by linear combination of weak classifiers. It performs drastically well but sometimes suffers the problem of overfitting. For the experimentation REPTree was used as its base classifier.
4. Bagging: It can do classification and regression based on a base classifier. The Base classifier used is the Decision stump and the bag-size percentage is 100%.
5. Nearest Neighbour: Nearest Neighbour is a non-parametric lazy learning classifier that assigns labels to a target sample on the basis of shortest distance with the training set. This classifier is simple and accurate yet computationally intensive.

Table 5. Comparison between the correctly classified instances and time taken to build in KDD and NSL-KDD test datasets using WEKA tool.

Classifier	Correctly classified instances (%)		Time taken to build (sec)		Kappa statistics (%)	
	KDD	NSL-KDD	KDD	NSL-KDD	KDD	NSL-KDD
J48	99.9	98.2	38.77	0.55	99.9	96.4
Naïve Bayes	92.7	80.7	29.01	0.08	88	62.4
AdaBoost	97.8	89.3	37.35	0.78	96.3	78.5
Bagging	99.9	98.4	128.6	1.78	99.9	96.8

The above table shows the correctly classified instances and the time taken to build the models to find the accuracy and false alarm rate in both KDD and NSL-KDD.

The First section consists of Introduction of what is intrusion detection. Second section is the description of KDD dataset and how many types of attacks are present in the dataset and how they are categorised. Third section consists of how to extract the features from dataset and how does machine learning algorithms works on the dataset like what is the accuracy and false alarm rate obtained on KDD dataset and results obtained while using WEKA tool for KDD and NSL-KDD. Next section consists of Conclusion and future work. Last section is the references used to understand the concepts.

4. CONCLUSION AND FUTURE WORK

In this survey we have introduced an overview of different machine learning algorithms for Intrusion Detection System (IDS) and different detection methodologies, and classifiers used in WEKA for KDD and NSL-KDD dataset. As per the studied of techniques suggested by various authors, the ways it can detect the intruder are presented here. The experiment results show that KNN is having high false rate and detection rate but AdaBoost algorithm has a very low false rate with high detection rate and run speed of algorithm is faster when compared with other algorithms. Our future work includes how we will find new types of weak classifiers to further improve the error rates and to find out is there any algorithm better than AdaBoost which is having high detection accuracy and false alarm rate.

REFERENCES

[1] Brandon Lokesak (December 4, 2008). "A Comparison Between Signature Based and Anomaly Based Intrusion Detection Systems" (PPT). www.iup.edu. Douligeris, Christos; Serpanos, Dimitrios N. (2007-02-09). Network Security: Current Status and Future Directions. John Wiley & Sons. ISBN 9780470099735.

[2] Rowayda, A. Sadek; M Sami, Soliman; Hagar, S Elsayed (November 2013). "Effective anomaly intrusion detection system based on neural network with indicator variable and rough set reduction". International Journal of Computer Science Issues (IJCSI). 10 (6).

[3] M. A. Maloof, Machine learning and data mining for computer security: Springer, 2006.

[4] J. M. Bonifacio, Jr., A. M. Cansian, A. C. P. L. F. De Carvalho, and E. S. Moreira, "Neural networks applied in intrusion detection systems," in Proc. IEEE Int. Joint Conf. Neural Netw., 1998, vol. 1, pp. 205–210.

[5] Shi-Jinn Horng, Ming-Yang Su, Yuan-Hsin Chen, TzongWann Kao, Rong-Jian Chen, Jui-Lin Lai, Citra Dwi Perkasa, A novel intrusion detection system based on hierarchical clustering and support vector machines, Journal of Expert systems with Applications, Vol. 38, 2011, 306-313

[6] Cheng Xiang, Png Chin Yong, Lim Swee Meng, Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees, Journal of Pattern Recognition Letters, Vol. 29, 2008, 918-924.

[7] Weiming Hu, Steve Maybank, "AdaBoost-Based Algorithm for Network Intrusion Detection". In IEEE transaction on systems, MAN, and CYBERNETICS, APRIL 2008.

[8] S. Sung, A.H. Mukkamala. "Identifying important features for intrusion detection using support vector machines and neural networks". In Proceedings of the Symposium on Applications and the Internet (SAINT), pp. 209–216. IEEE .

[9] H. Kayacik, A. Zincir-Heywood and M. Heywood. "Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets". In Proceedings of the Third Annual Conference on Privacy, Security and Trust (PST). 2005.

[10] C. Lee, S. Shin and J. Chung. "Network intrusion detection through genetic feature selection". In Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD), pp. 109–114. IEEE Computer Society,2006.

[11] M. Panda, M.R. Patra, "Semi-Naïve Bayesian method for network intrusion detection system", Neural information processing, Lecture Notes in Computer Science (Springer Link) 5863 (2009) 614–621.

[12] Sandeep Gurung, Mirnal Kanti Ghose, Aroj Subedi,"Deep Learning Approach on Network Intrusion Detection System using NSL-KDD Dataset", International Journal of Computer Network and Information Security(IJCNIS), Vol.11, No.3, pp.8-14, 2019.DOI: 10.5815/ijcnis.2019.03.02.

[13] Jawhar, M. M. T., & Mehrotra, M. (2010). Design network intrusion detection system using hybrid fuzzy neural network. International JournalofComputerScience and Security, 4(3), 285-294.

[14] Souza, P. V. C. (2018). Regularized Fuzzy Neural Networks for Pattern Classification Problems. International Journal of Applied Engineering Research, 13(5), 2985-2991.

.