Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 5, No. 1, May : 2023

SCALABLE AND HYBRID INTRUSION DETECTION SYSTEM USING CUSTOMIZED NEURAL NETWORKS

P. Anjani, P. Kanaka Durga Bhavani, P. Yasaswini UG Student,
C. Rashmi Assistant Professor,
Department of Information Technology, Malla Reddy Engineering College for Women (UGC-Autonomous), Maisammaguda, Secunderabad, Telangana, India

ABSTRACT

Machine learning techniques are being widely used to develop an intrusion detection system (IDS) for detecting and classifying cyber-attacks at the network-level and host-level in a timely and automatic manner. However, many challenges arise since malicious attacks are continually changing and are occurring in very large volumes requiring a scalable solution. In this project, customized neural network (CNN), a type of deep learning model is explored to develop a flexible and effective IDS to detect and classify unforeseen and unpredictable cyber-attacks. The continuous change in network behaviour and rapid evolution of attacks makes it necessary to evaluate various datasets which are generated over the years through static and dynamic approaches. This type of study facilitates identifying the best algorithm which can effectively work in detecting future cyber-attacks. A comprehensive evaluation of experiments of CNNs and other classical machine learning classifiers are shown on various publicly available benchmark malware datasets. Our CNN model learns the abstract and high dimensional feature representation of the IDS data by passing them into many hidden layers. Through rigorous experimental testing it is confirmed that CNNs perform well in comparison to the classical machine learning classifiers.

Keywords: Intrusion detection system, NSL-KDD, neural networks, machine learning

1. INTRODUCTION

Information and communications technology (ICT) systems and networks handle various sensitive user data that are prone to various attacks from both internal and external intruders. These attacks can be manual, machine generated, diverse and are gradually advancing in obfuscations resulting in undetected data breaches. For instance, the Yahoo data breach had caused a loss of $350M and Bitcoin breach resulted in a rough estimate of $70M loss. Such cyberattacks are constantly evolving with very sophisticated algorithms with the advancement of hardware, software, and network topologies including the recent developments in the Internet of Things (IoT). Malicious cyber-attacks pose serious security issues that demand the need for a novel, flexible and more reliable intrusion detection system (IDS). An IDS is a proactive intrusion detection tool used to detect and classify intrusions, attacks, or violations of the security policies automatically at network-level and host-level infrastructure in a timely manner. Based on intrusive behaviors, intrusion detection is classified into network-based intrusion detection system (NIDS) and host-based intrusion detection system (HIDS). An IDS system which uses network behaviour is called NIDS. The network behaviors are collected using network equipment via mirroring by networking devices, such as switches, routers, and network taps and analyzed in order to identify attacks and possible threats concealed within network traffic. An IDS system which uses system activities in the form of various log files running on the local host computer in order to detect attacks is called HIDS. The log files are collected via local sensors. While NIDS inspects each packet contents in network traffic flows, HIDS relies on the information of log files which includes sensors logs, system logs, software logs, file systems, disk resources, users account information and others of each system.

Many organizations use a hybrid of both NIDS and HIDS. Analysis of network traffic flows is done using misuse detection, anomaly detection and stateful protocol analysis. Misuse detection uses predefined signatures and filters to detect the attacks. It relies on human inputs to constantly update the signature database. This method is accurate in finding the known attacks but is completely ineffective in the case of unknown attacks. Anomaly detection uses heuristic mechanisms to find unknown malicious activities. In most of the scenarios, anomaly detection produces a high false positive rate. To combat this problem, most organizations use the combination of both misuse and anomaly detection in their commercial solution systems. Stateful protocol analysis is most powerful in comparison to the detection methods due to the fact that stateful protocol analysis acts on the network layer, application layer and transport layer. This uses the predefined vendors specification settings to detect the deviations of appropriate protocols and applications.

Though deep learning approaches are being considered more recently to enhance the intelligence of such intrusion detection techniques, there is a lack of study to benchmark such machine learning algorithms with publicly available datasets. The most common issues in the existing solutions based on machine learning models are: firstly, the models produce high false positive rate with wider range of attacks; secondly, the models are not generalizable as existing studies have mainly used only a single dataset to report the performance of the machine learning model; thirdly, the models studied so far have completely unseen today's huge network traffic; and finally the solutions are required to persevere today's rapidly increasing high-speed network size, speed and dynamics. These challenges form the prime motivation for this work with a research focus on evaluating the efficacy of various classical machine learning classifiers and deep neural networks (DNNs) applied to NIDS and HIDS. Overall, this work has made the following contributions to the cyber security domain:

- By combining both NIDS and HIDS collaboratively, an effective deep learning approach is proposed by modelling a deep neural network (DNN) to detect cyberattacks proactively. In this study, the efficacy of various classical machine learning algorithms and DNNs are evaluated on various NIDS and HIDS datasets in identifying whether network traffic behaviour is either normal or abnormal due to an attack that can be classified into corresponding attack categories.
- The advanced text representation methods of natural language processing (NLP) are explored with host-level events, i.e., system calls with the aim to capture the contextual and semantic similarity and to preserve the sequence information of system calls. The comparative performance of these methods is conducted with the NSL-KDD dataset.
- This study uses various benchmark datasets to conduct a comparative experimentation. This is mainly due to the reason that each data set suffers from various issues such as data corruptions, traffic variety, inconsistencies, out of date and contemporary attacks.

2. LITERATURE SURVEY

Ali et al. proposed a new intrusion detection system (IDS) based on the combination of fast learning network (FLN) and particle swarm optimization (PSO). The FLN is used to classify network traffic, and PSO is used to optimize the FLN's parameters. The proposed system was tested on the NSL-KDD dataset and achieved a high detection rate while maintaining a low false alarm rate. Shone et al. proposed a deep learning approach to network intrusion detection. The proposed system uses a deep belief network (DBN) to extract features from network traffic, and a support vector machine (SVM) classifier to classify the traffic as normal or anomalous. The system was tested on the UNSW-NB15 dataset and achieved a high detection rate while maintaining a low false alarm rate. The paper also compares the proposed system's performance with other state-of-the-art intrusion detection systems. Yin et al. proposed a deep learning approach using recurrent neural networks (RNNs) for intrusion detection. The authors use RNNs to model the temporal dependencies in network traffic and extract features that can distinguish between normal and anomalous network traffic.

In [4], authors proposed a deep learning-based intrusion detection system for smart meter communication networks. The authors use a combination of deep convolutional neural networks (CNNs) and long short-term memory (LSTM) networks to classify network traffic as either normal or malicious. Khan et al. proposed a two-stage deep learning model for efficient network intrusion detection. The authors use a combination of deep belief networks (DBNs) and convolutional neural networks (CNNs) to perform feature extraction and classification of network traffic. The proposed model achieves high accuracy with low computational cost. In [6], a multi-channel deep feature learning approach for intrusion detection is presented. The authors use a combination of autoencoders and convolutional neural networks to extract features from different network traffic channels. The proposed approach achieves high accuracy and robustness against adversarial attacks. Gu and Lu proposed an effective intrusion detection approach using support vector machines (SVMs) with naïve Bayes feature embedding. The authors use naïve Bayes to extract features from network traffic and embed them into an SVM classifier. The proposed approach achieves high accuracy and outperforms several other machine learning algorithms.

3. EXISTING SYSTEM

3.1 Support Vector Machine

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

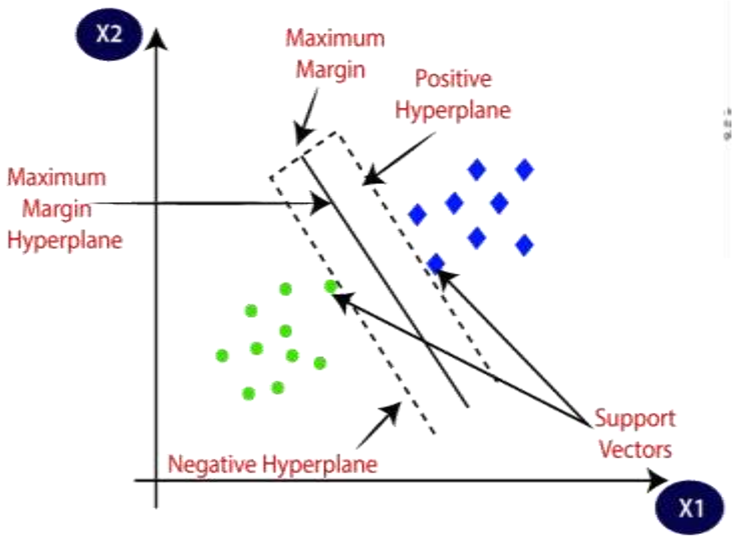


Figure 1. Analysis of SVM

3.2 Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

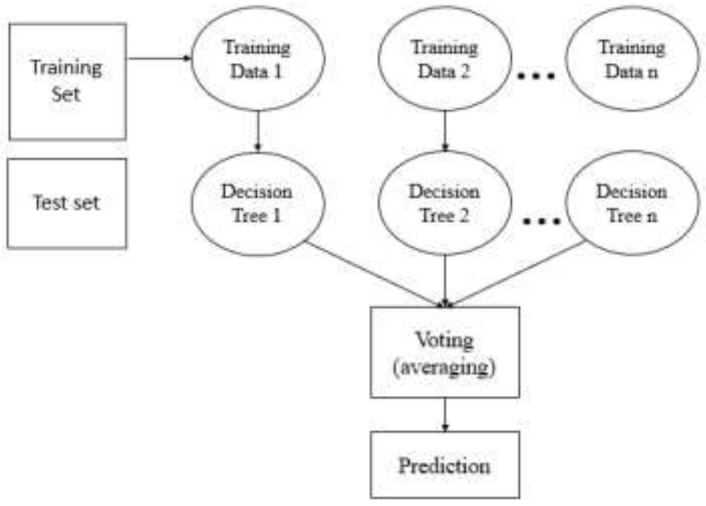


Figure 2. Random Forest algorithm

Random Forest algorithm

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

Important Features of Random Forest

- **Diversity**- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
- **Immune to the curse of dimensionality**- Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization**-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.
- **Train-Test split**- In a random forest we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability**- Stability arises because the result is based on majority voting/ averaging.

Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm.

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

Types of Ensembles

Before understanding the working of the random forest, we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model. Ensemble uses two types of methods:

Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest. Bagging, also known as Bootstrap Aggregation, is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

4. PROPOSED SYSTEM

Machine learning techniques are being widely used to develop an intrusion detection system (IDS) for detecting and classifying cyberattacks at the network-level and the host-level in a timely and automatic manner. However, many challenges arise since malicious attacks are continually changing and are occurring in very large volumes requiring a scalable solution. There are different Intrusion datasets available publicly for further research by cyber security community. However, no existing study has shown the detailed analysis of the performance of various machine learning algorithms on various publicly available datasets. Due to the dynamic nature of Intrusion with continuously changing attacking methods, the Intrusion datasets available publicly are to be updated systematically and benchmarked. This work evaluates the performance of various classical algorithms such as SVM, Random Forest and Deep Neural Network (DNN) etc to detect attacks on network using KDD, NSL datasets. The existing classical algorithms (SVM, Random Forest) are unable to predict dynamic (if attacker introduce new attacks with changes in attack parameter) attacks and needs to be trained in advance.

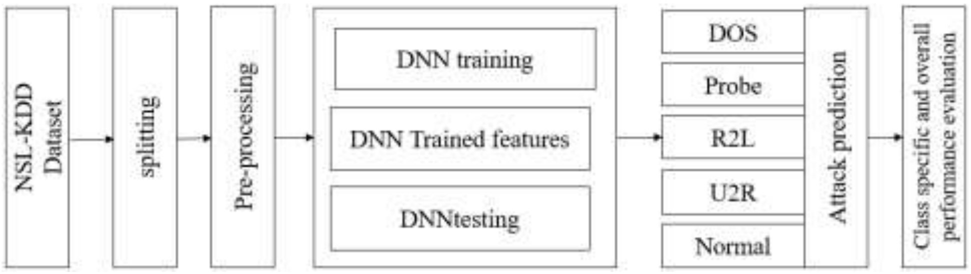


Figure 3: Proposed IDS methodology

Figure 1 shows the block diagram of proposed method. Initially, NSL-KDD dataset is split into 80% for training and 20% for testing. Then, dataset preprocessing operation is performed to normalize the entire dataset. Further, DNN classifier is used for prediction of attacks from test sample. The performance evaluation is carried out to show supremacy of the proposed method. The DNN is a famous algorithm which has high predicting ratio in all fields such as image processing, data classification etc. Therefore, DNN model is capable of detecting such attacks and to overcome from these attacking problems with dynamic attack signatures. The proposed DNN model contains multiple number of layers. The DNN algorithm keeps filtering training algorithm with hidden layer to form most accurate model to predict testing class. The common classes are Normal, Remote to user (R2L), Denial-of-Service (DOS), User to Root (U2R), Probe but in dataset we have other names, but all those names come under these classes.

3.1 Dataset

NSL-KDD is a data set suggested to solve some of the inherent problems of the KDD'99 data set. Although, this new version of the KDD data set still suffers from some of the problems discussed by McHugh and may not be a perfect representative of existing real networks, because of the lack of public data sets for network based IDSs, we believe it still can be applied as an effective benchmark data set to help researchers compare different intrusion detection methods.

Furthermore, the number of records in the NSL-KDD train and test sets are reasonable. This advantage makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research work will be consistent and comparable.

3.2 DNN

A typical architecture of DNN model for intrusion recognition is shown in Figure 4.5. DNNs are generally composed of three parts. Dense layer for feature extraction. The convergence layer, also known as the pooling layer, is mainly used for feature selection. The number of parameters is reduced by reducing the number of features. The full connection layer carries out the summary and output of the characteristics. A dense layer is consisting of a dense process and a nonlinear activation function ReLU.

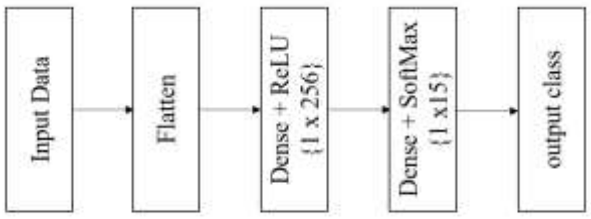


Figure 4. Proposed DNN model.

The leftmost data is the input layer, which the computer understands as the input of several matrices. Next is the dense layer, the activation function of which uses ReLU. The pooling layer has no activation function. The combination of dense and pooling layers can be constructed many times. The combination of dense layer and dense layer or dense layer and pool layer can be very flexibly, which is not limited when constructing the model. But the most common DNN is a combination of several dense layers and pooling layers. Finally, there is a full connection layer, which acts as a classifier and maps the learned feature representation to the sample label space.

DNN mainly solves the following two problems.

1) Problem of too many parameters: It is assumed that the size of the input test data is $50 * 50 * 3$. If placed in a fully connected feedforward network, there are 7500 mutually independent links to the hidden layer. And each link also corresponds to its unique weight parameter. With the increase of the number of layers, the size of the parameters also increases significantly. On the one hand, it will easily lead to the occurrence of over-fitting phenomenon. On the other hand, the neural network is too complex, which will seriously affect the training efficiency. In DNNs, the parameter sharing mechanism makes the same parameters used in multiple functions of a model, and each element of the denseal kernel will act on a specific position of each local input. The neural network only needs to learn a set of parameters and does not need to optimize learning for each parameter of each position.

2) Data stability: Data stability is the local invariant feature, which means that the natural data will not be affected by the scaling, translation, and rotation of the data size. Because in deep learning, data enhancement is generally needed to improve performance, and fully connected feedforward neural is difficult to ensure the local invariance of the data. This problem can be solved by dense operation in DNN.

ReLU layer: Networks those utilizes the rectifier operation for the hidden layers are cited as ReLU. This ReLU function $\mathcal{G}(\cdot)$ is a simple computation that returns the value given as input directly if the value of input is greater than zero else returns zero. This can be represented as mathematically using the function $max(\cdot)$ over the set of 0 and the input x as follows:

$$\mathcal{G}(x) = \max\{0, x\}$$

SoftMax classifier: Generally, SoftMax function is added at the end of the output since it is the place where the nodes are meet finally and thus, they can be classified as shown in Figure 5. Here, X is the input of all the models and the layers between X and Y are the hidden layers and the data is passed from X to all the layers and Received by Y. Suppose, we have 10 classes, and we predict for which class the given input belongs to. So, for this what we do is allot each class with a particular predicted output. Which means that we have 10 outputs corresponding to 10 different class and predict the class by the highest probability it has.

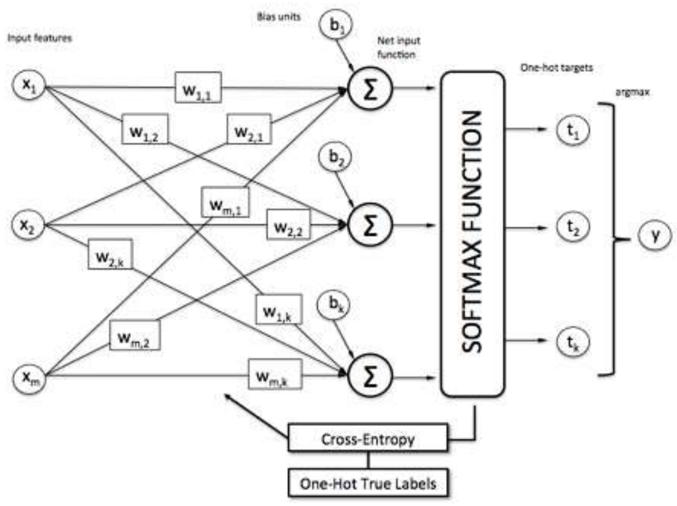


Figure 5. Intrusion class prediction using SoftMax classifier.

In Figure 6, and we must predict what is the object that is present in the test data. In the normal case, we predict whether the Intrusion is A. But in this case, we must predict what is the object that is present in the test data. This is the place where SoftMax comes in handy. As the model is already trained on some data. So, as soon as the test data is given, the model processes the test data, send it to the hidden layers and then finally send to SoftMax for classifying the test data. The SoftMax uses a One-Hot encoding Technique to calculate the cross-entropy loss and get the max. One-Hot Encoding is the technique that is used to categorize the data. In the previous example, if SoftMax predicts that the object is class A then the One-Hot Encoding for:

Class A will be [1 0 0]

Class B will be [0 1 0]

Class C will be [0 0 1]

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 5, No. 1, May : 2023

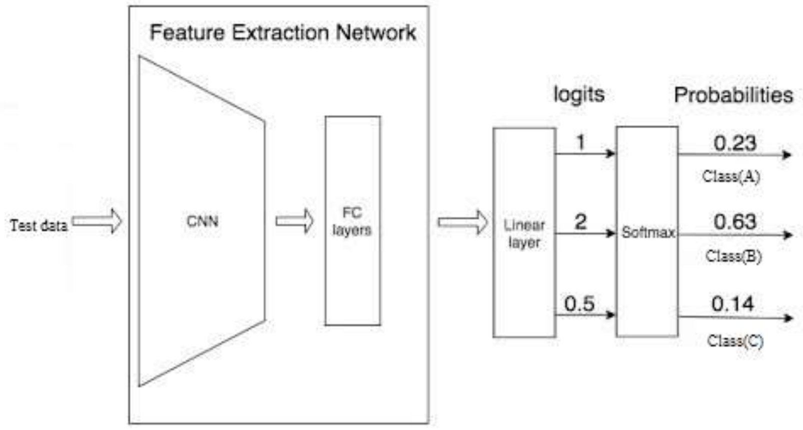


Figure 6. Example of SoftMax classifier.

From the Figure 7, we see that the predictions are occurred. But generally, we don't know the predictions. But the machine must choose the correct predicted object. So, for machine to identify an object correctly, it uses a function called cross-entropy function. So, we choose more similar value by using the below cross-entropy formula.

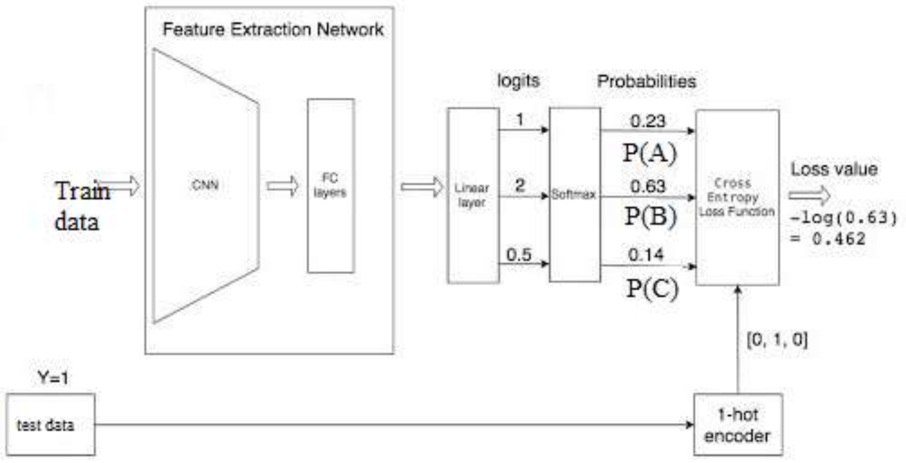


Figure 7. Example of SoftMax classifier with test data.

In the above example we see that 0.462 is the loss of the function for class specific classifier. In the same way, we find loss for remaining classifiers. The lowest the loss function, the better the prediction is. The mathematical representation for loss function can be represented as: -

$$LOSS = np.sum(-Y \ * \ np.log(Y\_pred))$$

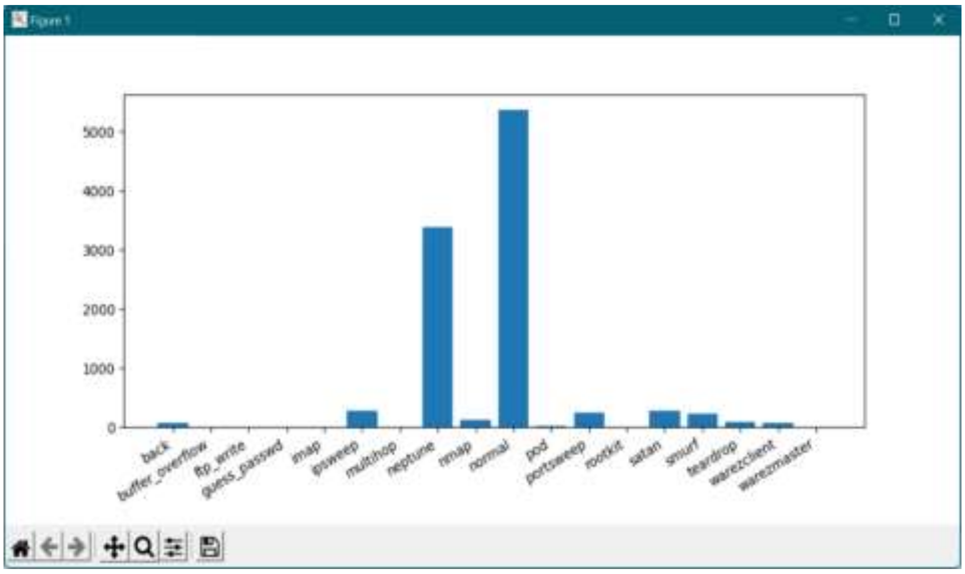
5. RESULT AND DISCUSSION

This project evaluates the performance of various classical algorithms such as SVM, and Random Forest to detect attacks on network using IDS datasets such as KDD, NSL but these classical algorithms unable to predict dynamic (if attacker introduce new attacks with changes in attack parameter) attacks and need to be trained in advance to detect such attacks. To overcome this problem, this work evaluates performance of customized neural network (CNN) model with dynamic attack signatures. The obtained detection accuracy of CNN shown to be better as compared to all classical algorithms.

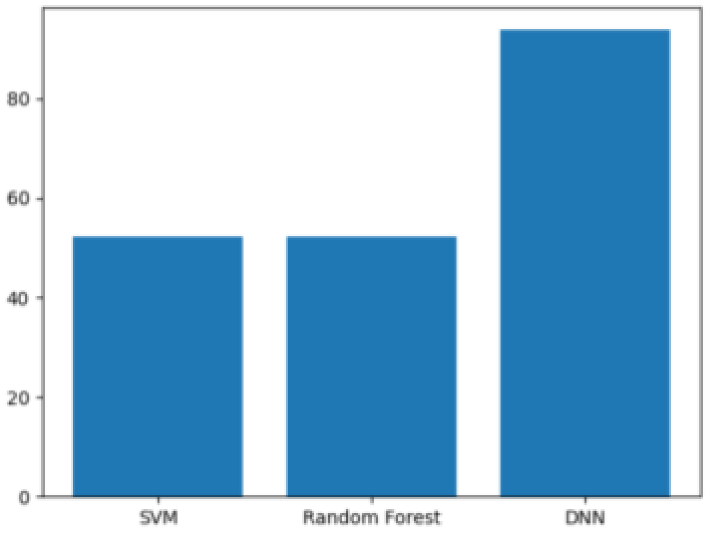
Here to implement this concept, KDD and NSL dataset combination is used with SVM, Random Forest and CNN model. CNN models keep filtering training algorithms with hidden layers to form the most accurate model to predict testing class. It is a famous model which has a high predicting ratio in all fields such as image processing, data classification etc. Below are the column names of dataset.

duration,protocol_type,service,flag,src_bytes,dst_bytes,land,wrong_fragment,urgent,hot,num_failed_logins,logged_in,num_compromised,root_shell,su_attempted,num_root,num_file_creations,num_shells,num_access_files,num_outbound_cmds,is_host_login,is_guest_login,count,srv_count,serror_rate,srv_serror_rate,rerror_rate,srv_rerror_rate,same_srv_rate,diff_srv_rate,srv_diff_host_rate,dst_host_count,dst_host_srv_count,dst_host_same_srv_rate,dst_host_diff_srv_rate,dst_host_same_src_port_rate,dst_host_srv_diff_host_rate,dst_host_serror_rate,dst_host_srv_serror_rate,dst_host_rerror_rate,dst_host_srv_rerror_rate,label

In the above dataset columns label is the name of attacks, all above comma separated names in bold format are the names of request signature.



In the above graph x-axis represents the attack name found in dataset and y-axis represents count of that attack type.



In the above graph x-axis represents algorithm name and y-axis represents accuracy and DNN is the proposed technique which got high accuracy compared to traditional algorithms such SVM and random forest.

6. CONCLUSION AND FUTURE SCOPE

This project proposed a hybrid intrusion detection system using a highly scalable framework on commodity hardware server which has the capability to analyze the network and host-level activities. The framework employed distributed deep learning model with DNNs for handling and analyzing very large-scale data in real time. The DNN model was chosen by comprehensively evaluating their performance in comparison to classical machine learning classifiers on various benchmark IDS datasets. In addition, we collected host-based and network-based features in real-time and employed the proposed DNN model for detecting attacks and intrusions. In all the cases, we observed that DNNs exceeded in performance when compared to the classical machine learning classifiers. Our proposed architecture is able to perform better than previously implemented classical machine learning classifiers in both HIDS and NIDS. To the best of our knowledge this is the only framework which has the capability to collect network-level and host-level activities in a distributed manner using DNNs to detect attack more accurately. The performance of the proposed framework can be further enhanced by adding a module for monitoring the DNS and BGP events in the networks. The execution time of the proposed system can be enhanced by adding more nodes to the existing cluster. In addition, the proposed system does not give detailed information on the structure and characteristics of the malware. Overall, the performance can be further improved by training complex DNNs architectures on advanced hardware through distributed approach. Due to extensive computational cost associated with complex DNNs architectures, they were not trained in this research using the benchmark IDS datasets. This will be an important task in an adversarial environment and is considered as one of the significant directions for future work.

REFERENCES

[1] M. H. Ali, B. A. D. Al Mohammed, A. Ismail and M. F. Zolkipli, "A New Intrusion Detection System Based on Fast Learning Network and Particle Swarm Optimization", IEEE Access, vol. 6, pp. 20255-20261, 2018.

[2] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection", IEEE Trans. Emerg. Top. Comput. Intell, vol. 2, no. 1, pp. 41-50, Feb. 2018.

[3] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks", IEEE Access, vol. 5, pp. 21954-21961, 2017.

[4] R. Vijayanand, D. Devaraj, and B. Kannapiran, "A Novel Deep Learning Based Intrusion Detection System for Smart Meter Communication Network", 2019 IEEE International Conference on Intelligent Techniques in Control Optimization and Signal Processing (INCOS), pp. 1-3, Apr. 2019.

[5] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "TSDL: A Two-Stage Deep Learning Model for Efficient Network Intrusion Detection", IEEE Access, vol. 7, pp. 30373-30385, 2019.

[6] G. Andresini, A. Appice, N. D. Mauro, C. Loglisci, and D. Malerba, "Multi-Channel Deep Feature Learning for Intrusion Detection", IEEE Access, vol. 8, pp. 53346-53359, 2020.

[7] J. Gu and S. Lu, "An effective intrusion detection approach using SVM with naïve Bayes feature embedding", Computers & Security, vol. 103, pp. 102158, Apr. 2021.