ENSEMBLE TECHNIQUE BASED INTRUSION DETECTION SYSTEM

L K Suresh Kumar
University College of Engineering, Osmania University, India

ABSTRACT

Traditional machine learning-based Intrusion detection systems often consider using single algorithm for IDS classification, which leads to handling bigdata, higher dimensional data difficult. And also the accuracy and other evaluation metrics achieved will not be that good to use those models in modern real-time environments, where even the smallest inaccuracy in detecting an intrusion will cost plenty. In this paper, the proposal of the use of stacking ensemble method as a classifier in IDS is discussed. Proposed framework is evaluated on the CICIDS2017 dataset. Related work on IDS using ML techniques, data pre-processing, the algorithms used in the classification module and the experimental results are presented in this paper. The experimental results achieved are high compared to other previously done works.

KEYWORDS: Ensemble methods**,** Intrusion detection systems, Machine learning.

1.INTRODUCTION

In the recent years the use of computers and therefore Internet has widespread. Many advanced technologies like cloud computing, IOT (Internet of things), huge amounts of accessible data, online transactions etc. are becoming vulnerable to cyber-attacks. There has been a significant increase in the number of malicious activities that are happening in these complex environments. The need to protect these networks from malicious activities is increasing day-by-day; this is where the Network security comes into play.

Due to different types of Intrusions, cyber-attacks, huge amount of monetary loss is happening. A survey was conducted in 2018 to calculate the amount of money lost in different countries and is shown in the following graphical demonstration [1] *(fig-1).*

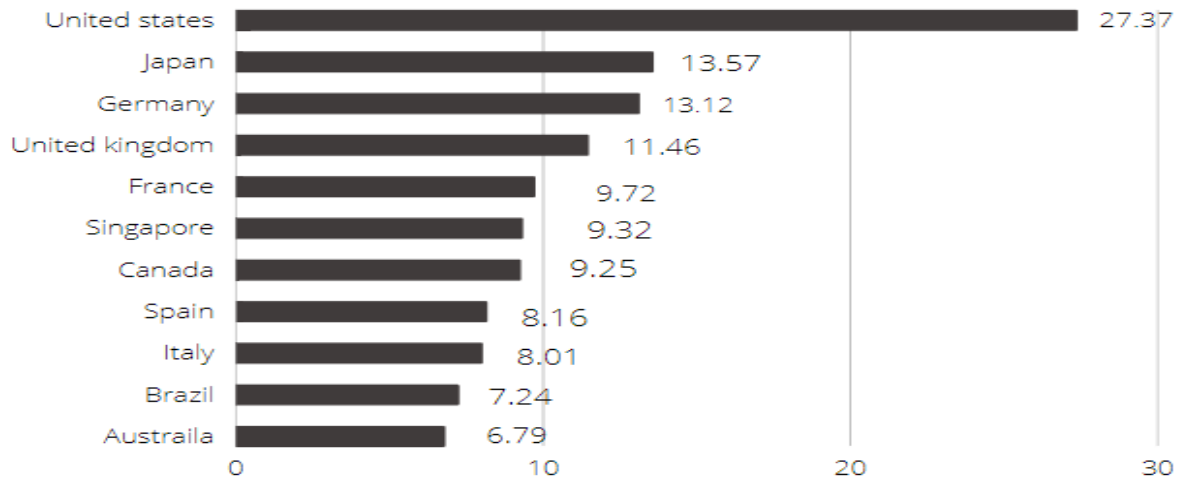

Fig-1: Average annualized cost of cyber-attacks on companies in selected countries in 2018(in million U.S. dollars).

Network security consists of three major components.1. Firewalls; 2. Antivirus software; 3.**IDS** (Intrusion Detection Systems). *Basic Terminology:* Intruder - Person trying to gain unauthorized access to a network, with some criminal intention Intrusion - Process of gaining unauthorized access. *There are 2 types of intruders:* Outside intruder - Also called masquerader, no authorized access. Inside Intruder - Misfeasor, has access to some privileges, misuses the information like selling to third party. To protect the networks from these intrusions, intrusion detection systems come into play. Intrusion Detection System (IDS) is an important tool use in cyber security to monitor and determine intrusion attack. IDS continuously monitors the system and network traffic (data packets) and checks for malicious content. If anything, suspicious is caught, then IDS sends alert signal to the system or network administrator.

IDS based on detection approach: 1. Signature-based: In this approach the known intrusion attack patterns are stored in a database, if a match occurs between the current system data and the database then IDS identifies it as an attack (malicious activity). This approach provides a fast and accurate detection. However, the slight drawback is this approach requires periodic(continuous) update of the database with new intrusion patterns. *2. Anomaly-based:* Also known as behavior-based, in this approach the IDS determines an attack when the system operates out of norm. This approach can detect both known and unknown attacks. The drawback of this approach is low accuracy with high FAR (False Alarm Rates).

IDS based on deployment type: 1. Network Intrusion Detection System (NIDS): Network IDS Monitors, captures, and analyzes network traffic, helps to detect malicious activities by identifying suspicious patterns in the incoming data packets. Drawback is that NIDS analysis is somewhat difficult in a busy network. *2.Host Intrusion Detection System (HIDS):* Installed on individual host or device on network. It monitors data packets from the device only and will alert the admin if suspicious activity is detected. Takes snapshot of the Existing System and compares it with previous System (Clean System) and identifies suspicious activity in the host machine.

Machine learning (ML) techniques can be used in Intrusion detection systems in helping predict the class of an attack. This can be achieved by using various classification methods available in machine learning, which learn and extract useful information from the features of the data. Based on the approach of the training of the classifier, IDS can be categorized into two types *1. supervised learning:* The model is trained using huge amounts of well-labelled data as much as possible to predict the label of the test data (outside training data). *2. Unsupervised learning:* The model is trained using unlabeled data to understand and extract the structural knowledge of the data present thus predicting which labels of the testing data. From the recent past years researches tried and are trying to apply various machine learning algorithms for the purpose of classification. These algorithms include SVM (support vector machine) [3], KNN (k-nearest neighbor) [4], RF (random forest) [2], Neural networks like CNN (convolutional neural network), ANN (artificial neural network) [5], DNN (deep neural network).

Traditional ML algorithms that are used in IDS classification often consider only single algorithm, which is the reason for low detection rates, high dimensional data. To overcome this issue that is to improve the accuracy of detection, to lower false alarm rates we are using stacking ensemble method for the classification of the data. Considering that stacking models

stack multiple classifiers and build a more accurate model compared to the individual classifiers we have tried implementing it on the [6] CICIDS2017 dataset. we have used Decision tree, random forest, k nearest neighbour algorithms as base classifiers and meta classifier is Logistic regression.

In summary, the paper's main contributions are as follows:

1. We evaluated our stacking model using the CICIDS2017 dataset. Compared with different methods in the related work using the same dataset and also compares the results obtained by using the individual classifiers used in base classifier.
2. We found out that the accuracy, other classification metrics have improved by using the stacked model.

With the increase in technology and the use of internet, the possibility of network intrusions is skyrocketing day-by-day. To detect and prevent these attacks from happening and saving important private information from being exploited by the intruders and building an efficient IDS that suits modern requirements is the main motivation behind this work.

Remainder of the paper is organised as follows: Section 2 states the related work done on IDS using machine learning techniques and discusses their results obtained. Section 3 briefly describes the proposed methodology and dataset used to evaluate the stacking classifier and some of the pre-processing required for the subsequent experiments. In Section 4, we evaluate the performance of our approach based on the experimental results and compare it with some previous work. Section 5 summarizes and discusses future work.

2.RELATED WORK

This section reviews the related work done on IDS using machine learning techniques.

Elbasiony et al. [7] presented a hybrid detection framework based on data mining classification, clustering techniques. They used random forest algorithm along with a weighted k-means algorithm on the KDD'99 dataset and achieved a detection rate of 98.3%. Yao et al. [8] proposed a hybrid multilevel data mining IDS framework which can detect known, unknown attacks. They tested their model on KDDCUP99 dataset and achieved a 96.70% accuracy. Min et al. [9] proposed an autoencoder-based framework for network intrusion detection framework used clustering loss and reconstruction loss combinedly trained by the unsupervised clustering module and autoencoder to obtain a more cluster-friendly feature representation for better clustering results.

MR karbir et al. [10] eliminated irrelevant features using genetic algorithm feature selection techniques. They used Bayesian methos as base classifier for attack prediction on the NSL-KDD dataset, achieved an accuracy of 98.265%. Gu et al. [11] obtained a new transformed train data by transforming actual features into logarithmic marginal density rate as SVM is highly dependent on the dataset size. They used SVM ensemble on the NSL-KDD dataset and achieved an accuracy of 99.41%. Ahmim et al. [12] proposed three different classifier-based hierarchical intrusion detection systems based on reduced error pruning (REP) tree, JRip algorithm, and Forest PA. First and second methods took the features of the dataset as input and classified the network traffic as attack/benign; the third classifier took the output of the first classifier, second classifier, and the features of the initial dataset were considered as input. They used the CICIDS2017 dataset and achieved 96.665% accuracy.

Kumar et al. [13] used random forest regression technique for selecting a subset of attributes from the original set, and then the selected set of important features were used in the trained classifier to solve the problem of low detection rate. Authors in [14] used the self-learning concept to train deep neural networks to perform feature extraction by pretraining the network, combining raw and extracted features to train sparse auto encoder. Nathan et al. [15] designed a stacked nonsymmetric deep autoencoder for unsupervised feature learning, using the random forest as a classifier, while also reducing the training time required for training. The model was evaluated using the KDD Cup'99 and NSL-KDD datasets, which showed better performance in terms of accuracy for binary classification. Dimensionality feature reduction is not considered in this design which can be seen as a drawback.

M. Al-Qatf [16] used sparse autoencoder to obtain reconstructed new feature representations, using SVM for classification and reducing the training and testing time, and the method was evaluated on the NSL-KDD dataset and achieved 99.416% accuracy and 99.396% accuracy for binary and multiclass classification, respectively. Abdulhammed et al. [17] used auto encoder and principal component analysis (PCA) to reduce the feature dimensionality of the CICIDS2017 dataset. The low-dimensional features generated by the two techniques were used to construct various classifiers, such as RF, Bayesian networks, linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA) to design IDS. They converted the source IP address and destination IP address into integer formats for model training. But some of the features of the dataset (e.g., Source IP and Timestamp) mentioned in the literature above will make the model adapt to a specific dataset when training the model using the CICIDS2017 dataset in separate experiments in [18] on whether to use IPs as features for training shows a 3% difference.

Yin et al. [19] proposed a Generative adversarial network-based model (GIDS) to improve the performance and generalization capability of classifiers, which used adversarial training to enhance the classifiers and generated false label samples continuously using a generative model to help the classifiers in increasing their detection rate. Vlajic et al. [26] analysed contamination based on anomaly detection in the adaptive IDS, using autoencoder reconstruction error as a metric for anomaly detection, using training time and test temporal metrics to evaluate the performance of the model. They used NSL-KDD datasets for evaluation, and maintaining a stable detection situation while reducing the contamination level of the training dataset to less than 2% compared with a principal component analysis-based IDS.

3.PROPOSED METHODOLOGY

In this section we will introduce the proposed methodology, classification algorithm used and its workflow. The proposal talks about the use of stacking ensemble method, which is one of the ensemble learning techniques.

Ensemble learning: It is a technique in machine learning which uses the concept of combining multiple classifiers rather than sticking to just a single algorithm. It uses multiple machine learning algorithms to obtain a better result than that could be obtained from the constituent algorithms alone. Ensemble learning is mainly of three types: 1. Bagging; 2. Boosting;3. Stacking.

Bagging: It is a short form of bootstrap aggregation. Bootstrap is a sampling technique used to obtain different samples from the training data using replacement procedure. Aggregation refers to a technique that combines all possible outcomes of the prediction and randomizes the outcome. Hence many weak models are combined to form a better model. *Ex:* Random Forest. *Boosting:* Boosting is an ensemble technique that learns from previous predictor errors in order to make better predictions in the future. The technique combines several weak base learners to form one strong learner, significantly improving model predictability. *Ex*: XG Boost, Gradient Boosting etc. *Stacking:* Stacking also known as stacked generalization. This technique works by allowing a training algorithm to combine the predictions of several other similar learning algorithms. Stacking has been used successfully in regression, distance learning, and classification. It can also be used to calculate the error rate during bagging.

3.1. Classification Algorithm: In the classification section we have used a stacked model as a classifier. Our stacked classifier contains Decision tree, Random Forest, K nearest neighbours as base classifiers (BC) and we have applied Logistic regression as the meta-classifier (MC). *Algorithms used in Base classification: Decision Tree (DT*): Decision Tree Algorithm belongs to the family of supervised algorithms. It can be used for solving regression and classification problems. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node. A decision tree simply asks a question and based on the answer (yes/no) It further splits the tree into sub-trees. The DT algorithm automatically selects the best features for building a tree (using ATTRIBUTE SELECTION MEASURES Example (Information gain)/ (Gini index)) and then perform pruning operation to remove irrelevant branches from the tree to avoid the over-fitting. The most common DT models are CART, C4.5, and ID3. A very general decision tree representation is shown in fig-2.

Usage (in this paper): Classification purpose.

Advantages. 1. It can capture nonlinear relationships: They can be used to classify non-linearly separable data. *2.* It does not require any transformation of the features if we are dealing with nonlinear data because decision trees do not take multiple weighted combinations into account simultaneously. 3. They are very fast when compared to KNN and other classification algorithms. 4. The date type of DT can be numerical or categorical, or Boolean. 5. Normalization is not required. 6.There is less requirement of data cleaning compared to other algorithms.

[21] Classification and regression tree (CART) is chosen as the DT algorithm for this experiment (see [29] for details of CART). CART is a binary tree, which uses the binary segmentation method to cut the data into two parts to enter the left subtree and right subtree, respectively. And, each non-leaf node has two children, so CART has one more leaf node than non-leaves. In CART classification, the Gini index is used to select the best features for data partitioning, and Gini describes purity, the smaller the value, the higher the purity and the better the features. We select the attribute that minimizes the Gini index after division in

the candidate set as the optimal sub attribute; each iteration in CART reduces the Gini

Input: training dataset
Output: tree structured classifier
S: number of training samples
M: number of features. m: number of features input (m << M)
N: number of trees generated
If the tree to be generated is less than N,
Step 1: from the S training samples, take samples S times in a way with a put-back sampling to form a training set
Step 2: use unselected samples to make predictions and evaluate their errors
Step 3: for each node, m features are randomly selected
Step 4: according to these m features, calculate the best split method
Step 5: grow to be largest extent possible without pruning
Return tree structured classifier

coefficient. For sample set D, the number is |D|. Suppose there are K classes and the kth class is |Ck|, then the Gini index expression for sample set D is

Input: training dataset D
Output: CART
N: threshold of the number of samples in node. n: number of samples in the node
G: Gini index threshold for D
Gini (D): Gini index of D
Based on D, starting from the root node, if n < N or Gini (D) < G or no more features, recursively perform the following operations on each node to construct a binary tree
For each feature A, for each of its possible value a, the split will be into D1 and D2 based on whether the test for A= a, and use equation (2) to calculate Gini (D, A)
Among all possible features A and all its possible segmentation points a, the feature with the smallest Gini index and its corresponding segmentation point are selected as the optimal feature and the optimal segmentation point
Generate two sub nodes from the current node and assign the training dataset to the two sub nodes according to the features
Return CART

$$\text{Gini (D)} = 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D|}\right)^2 \quad (1)$$

[21] For sample D, the number is |D|, which divides D into |D1|, |D2|, . . ., |Dn| n parts according to certain value a of feature A; then, under the condition of feature A, the Gini index expression for sample set D is

$$\text{Gini (D, A)} = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \text{Gini } (D_i) \quad (2)$$

Algorithm 1 gives the basic process for CART creation, which involves following variable selection criteria and splitting criteria to grow the tree until the stopping criteria is met. Algorithm 2 gives the normal tree structured classifier generation process.

[21]

Algorithm 1: CART creation process.

[21]

Algorithm 2: The tree classifier generation process.

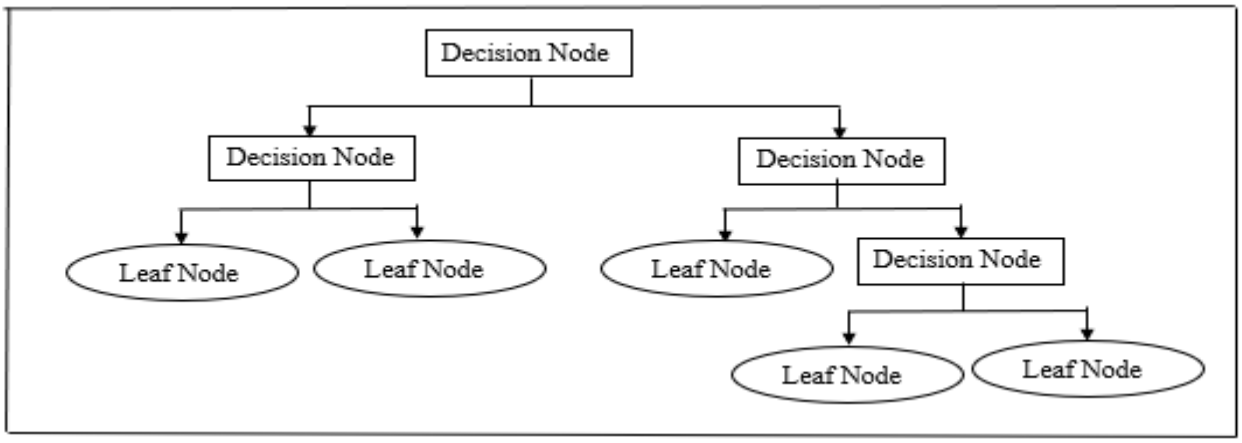

Fig-2: General representation of a decision tree.

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. Resampling using the bootstrap approach is used for the creation of each tree in the forest. Also, on each node split a subset of features is selected randomly and the selection of the split variable occurs over this subset. The predicted value is the majority vote, for classification, and the average, for regressions.

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: *1.*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. *2.* The predictions from each tree must have very low correlations.

Advantages:

1. Random Forest is based on the bagging algorithm and uses Ensemble Learning technique. It creates as many trees on the subset of the data and combines the output of all the trees. In this way it reduces overfitting problem in decision trees and also reduces the variance and therefore improves the accuracy. *2***.** Random Forest can be used to solve both classification as well as regression problems. *3***.** Random Forest works well with both categorical and continuous variables. *4*. Random Forest can automatically handle missing values. *5***.** *No feature scaling required:* standardization and normalization are not required in case of Random Forest as it uses rule-based approach instead of distance calculation. *6***.** *Handles non-linear parameters efficiently:* Non-linear parameters don't affect the performance of a Random Forest unlike curve-based algorithms. So, if there is high non-linearity between the independent variables, Random Forest may outperform as compared to other curve-based

algorithms. *7.* Random Forest is usually robust to outliers and can handle them automatically. *8.* Random Forest algorithm is very stable. Even if a new data point is introduced in the dataset, the overall algorithm is not affected much since the new data may impact one tree, but it is very hard for it to impact all the trees.

K Nearest Neighbours: KNN is a supervised learning algorithm and is one of the simplest machine learning algorithms. KNN algorithm assumes the similarity between the new data (testing data) and available data (training data) and arranges/ classifies the new data into the class that is most similar to the available classes. The parameter K in KNN is the number of

Step-1: Select the parameter K which is the number of nearest neighbours the algorithm should consider while classifying the data.
Step-2: Calculate the Euclidean distance of k number of neighbours
Step-3: Take the K nearest neighbours as per the calculated Euclidean distance.
Step-4: Among these K neighbours, count the number of the data points in each category.
Step-5: Assign the new datapoints to that category for which the number of the neighbour is maximum.
Step-6: KNN Model is ready.

Algorithm 3: KNN Classifier

nearest data points that the algorithm needs to consider while classifying a new datapoint and is passed by the user. KNN algorithm can be used for both classification and regression. But, is mostly used for classification as KNN classifies the new datapoint using the similarity of the data with its previous data. K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs the action on the dataset. KNN algorithm stores the dataset at training phase and when it gets new data, then it classifies that data into a category that is much similar to the new data. Algorithm 3 shows the KNN procedure.

We have used Logistic Regression as meta learner. *Logistic Regression:* [23] Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the class of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function (Sigmoid function), which predicts two maximum values (0 or 1). Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. *Logistic Function (Sigmoid Function):* The sigmoid function is a mathematical function used to map the predicted values to probabilities. It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go

beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function. In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0. *Assumptions for Logistic Regression:* The dependent variable must be categorical in nature. The independent variable should not have multi-collinearity.

Logistic Regression Equation: The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

We know the equation of the straight line can be written as:

$$Y=b0 + b1x1 + b2x2 + \ldots + bnxn \quad (3)$$

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$y/1-y; \quad 0 \text{ for } y=0, \text{ and infinity for } y=1 \quad (4)$$

But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$Log[y/1-y] = b0 + b1x1 + b2x2 + \ldots + bnxn \quad (5)$$

The above equation is the final equation for Logistic Regression. Flow chart representing the training procedure of stacking classifier is shown in fig-7 where Logistic regression is chosen as the meta-classifier.

4.RESULTS AND DISCUSSIONS

Classification performance is the most important aspect of IDS. In this section, we focus on evaluating classification performance and the experimental validation of our proposed approach. We first described in detail the CICIDS2017 dataset used for the experiments and the data pre-processing done, gave the evaluation metrics and the experimental environment used for the evaluation, and compared and analysed the experimental results.

The datasets like NSL-KDD, KDDCUP'99 are old and do not match the current trends in network security. That is why we have chosen the CICIDS2017 dataset which is a latest dataset consisting of variety of attack data which suits more to the current trends. [21] This dataset constructs the abstract behaviour of 25 users based on HTTP, HTTPS, FTP, SSH, and e-mail protocols to simulate the real network environment accurately.

4.1. CICIDS2017 Dataset: [21] The CICIDS2017 dataset was collected based on real traces of normal and malicious activity in the network traffic. The total number of records in the dataset is 2,830,743. The normal activity traffic contains 2,273,097 records (about 80.3% of the data) and malicious behaviour traffic records 557,646 records (about 19.7% of the data). The distribution of each tag set for the entire dataset is given in Figure 3. The CICIDS2017 dataset is a labelled dataset with 85 features that contain class labels (last column) corresponding to the traffic state. These features are listed in Table 1. The CICIDS2017

dataset is one of the unique datasets containing the latest attacks, and we used the Wednesday-working hours dataset to test and evaluate our proposed method performance. Since, CICIDS2017 dataset is a huge dataset with almost 28 and half lakh records, keeping in mind the amount of training time it takes for a dataset of such size, we have considered only 50% of it. Flow chart of the training procedure of stacking classifier is shown in Fig-3. A detailed visualisation of the chosen dataset is shown in fig-4 and the list of features is shown in [24] fig-5.

4.2. Data pre-processing: Some regular operations were performed on the chosen dataset like handling null values, handling infinite values and then we have encoded the objects (character data) into numeric types using label encoder. We have split the data using train_test_split in the ratio 8:2, out of which training data is 80% and testing data is 20%.

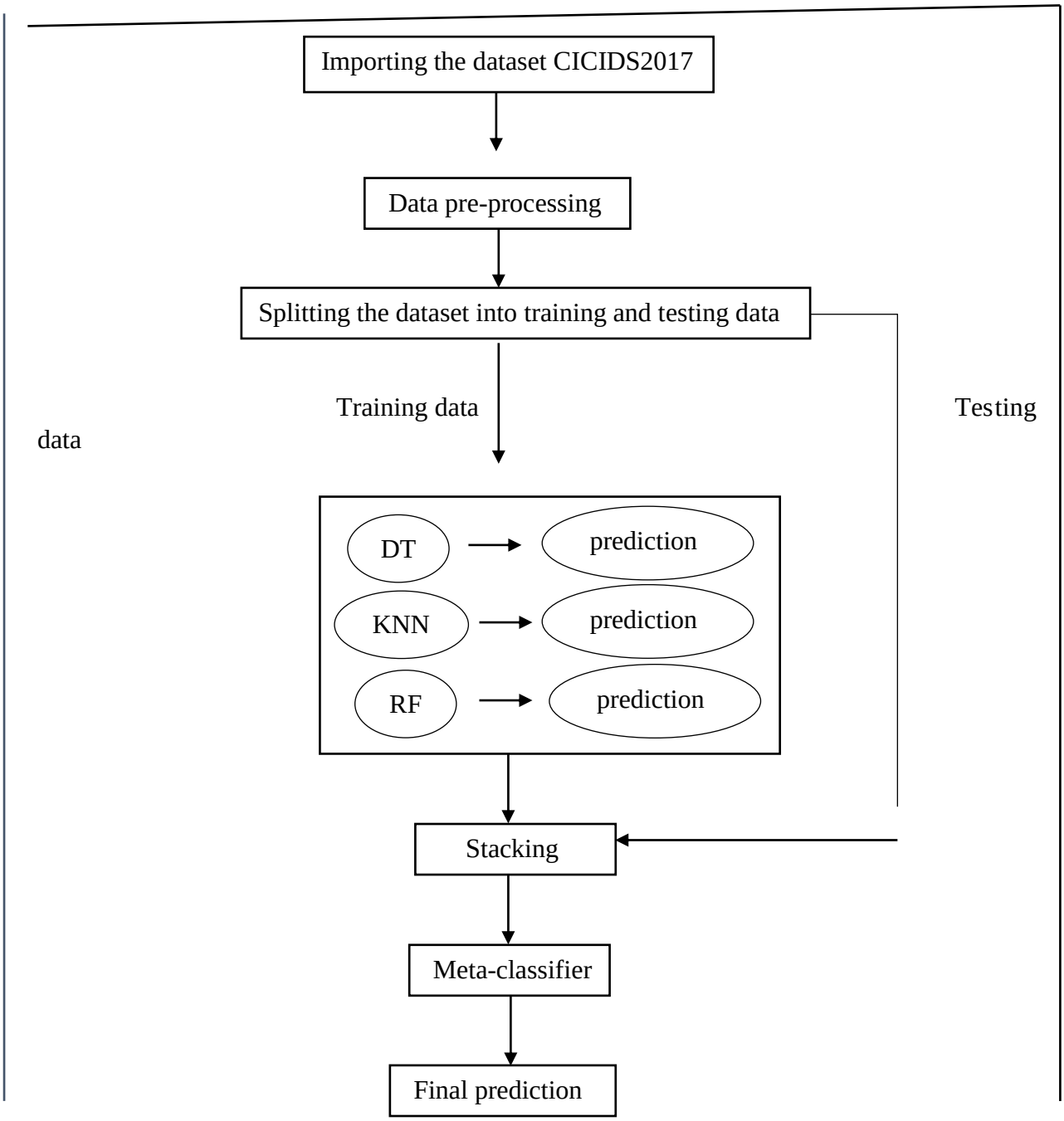

Fig-3: Training procedure of stacking classifier

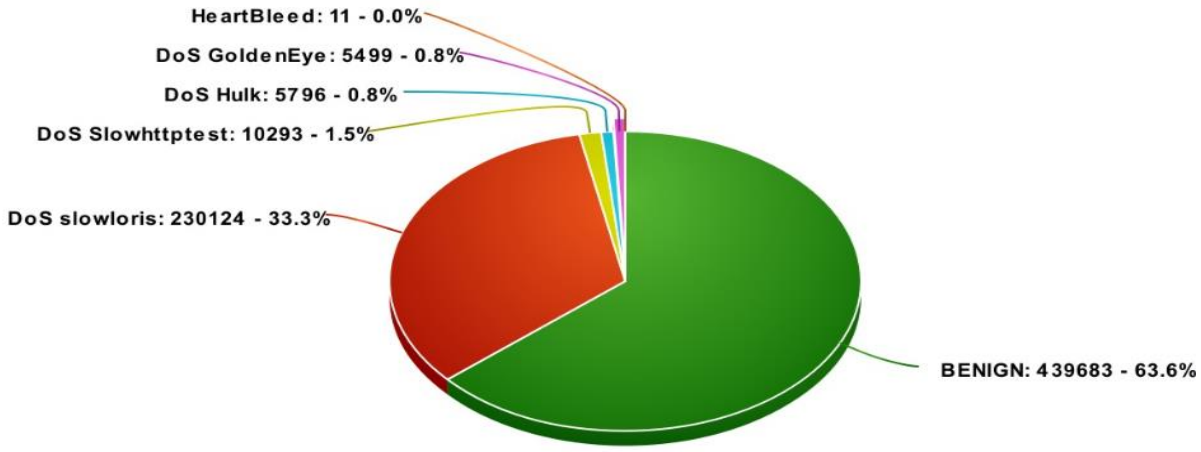

Fig-4: Distribution of labels in chosen dataset.

No.	Feature	No.	Feature	No.	Feature
1	Flow ID	29	Fwd IAT Std	57	ECE Flag Count
2	Source IP	30	Fwd IAT Max	58	Down/Up Ratio
3	Source Port	31	Fwd IAT Min	59	Average Packet Size
4	Destination IP	32	Bwd IAT Total	60	Avg Fwd Segment Size
5	Destination Port	33	Bwd IAT Mean	61	Avg Bwd Segment Size
6	Protocol	34	Bwd IAT Std	62	Fwd Avg Bytes/Bulk
7	Time stamp	35	Bwd IAT Max	63	Fwd Avg Packets/Bulk
8	Flow Duration	36	Bwd IAT Min	64	Fwd Avg Bulk Rate
9	Total Fwd Packets	37	Fwd PSH Flags	65	Bwd Avg Bytes/Bulk
10	Total Backward Packets	38	Bwd PSH Flags	66	Bwd Avg Packets/Bulk
11	Total Length of Fwd Pck	39	Fwd URG Flags	67	Bwd Avg Bulk Rate
12	Total Length of Bwd Pck	40	Bwd URG Flags	68	Subflow Fwd Packets
13	Fwd Packet Length Max	41	Fwd Header Length	69	Subflow Fwd Bytes
14	Fwd Packet Length Min	42	Bwd Header Length	70	Subflow Bwd Packets
15	Fwd Pck Length Mean	43	Fwd Packets/s	71	Subflow Bwd Bytes
16	Fwd Packet Length Std	44	Bwd Packets/s	72	Init_Win_bytes_fwd
17	Bwd Packet Length Max	45	Min Packet Length	73	Act_data_pkt_fwd
18	Bwd Packet Length Min	46	Max Packet Length	74	Min_seg_size_fwd
19	Bwd Packet Length Mean	47	Packet Length Mean	75	Active Mean
20	Bwd Packet Length Std	48	Packet Length Std	76	Active Std
21	Flow Bytes/s	49	Packet Len. Variance	77	Active Max
22	Flow Packets/s	50	FIN Flag Count	78	Active Min
23	Flow IAT Mean	51	SYN Flag Count	79	Idle Mean
24	Flow IAT Std	52	RST Flag Count	80	Idle Packet
25	Flow IAT Max	53	PSH Flag Count	81	Idle Std
26	Flow IAT Min	54	ACK Flag Count	82	Idle Max
27	Fwd IAT Total	55	URG Flag Count	83	Idle Min
28	Fwd IAT Mean	56	CWE Flag Count	84	Label

Fig-5: List of features in CICIDS2017 dataset.

4.3. Evaluation Metrics. The detection module as the main function of the framework needs to evaluate the performance of the framework accurately; we used four metrics to evaluate the performance of the classification module: accuracy, recall, precision, and F1-score, which are widely used to evaluate the performance of classification algorithms used in intrusion detection techniques. Their formulas are as follows.

Accuracy (Acc): it is defined as the ratio of correctly classified samples to the total number of samples, which represents the overall performance of the model.

$$Acc = (TP + TN) / (TP + TN + FP + FN) \quad (6)$$

Recall (Re) or detection rate (DR): it is defined as the ratio of the number of samples correctly classified into a certain class to the actual number of samples of this class.

$$Re = DR = (TP) / (TP + FN) \quad (7)$$

Precision (Pr): it is defined as the ratio of the number of samples correctly identified as a category to the number of samples identified as such.

$$Pr = (TP) / (TP + FP) \quad (8)$$

F1-score (F1) or F-measure (FM): it is defined as the harmonic mean of Precision and Recall.

$$F1 = FM = (2 * Pr * Re) / (Pr + Re) \quad (9)$$

All these evaluation metrics are derived from the four values found in the confusion matrix (Fig-7), and for each type of sample, TP is the number of samples correctly classified as that type, TN is the number of samples correctly classified as not that type, FP is the number of samples misclassified as that type, and FN is the number of samples misclassified as not that type.

Our experiment was conducted to investigate the performance of our proposed framework in multiclass classification. Fig-6 shows the performance of the algorithms used in base classifier (DT, RF, KNN) and different parameters (for RF we checked with different number of decision trees i.e., n_estimators) on the values of the four evaluation metrics in multiclass classification. Fig-7 shows the confusion matrix of our stacked model. From these figures we can see that different parameter algorithms (except DT) have subtle differences in performance against different attacks and proposed method showed higher performance in all the four evaluation metrics. A better algorithm can detect more attacks in shorter time and more adapt to the modern Internet; we choose the appropriate algorithm based on the evaluation metrics and our stacked model approach is a suitable classifier to a greater extent. Table 1 provides a detailed summary of the proposed framework in terms of Precision, Recall, Accuracy, and F1-score, respectively. Attack samples like Heartbleed have too few records than other attacks but all of them were identified. We also validated the superiority of our model by comparing its performance with the classification approaches used in related research. Table 2 is a comparison between the related work using multiclass classification results. Our proposed method achieved an accuracy of 98.95%.

Table 1: Evaluation results of various classifiers on the data.

Classifier	Precision	Recall	Accuracy	f1_score
RF-10	0.9693	0.9625	0.9624	0.9621
RF-20	0.9689	0.9693	0.9693	0.9693
RF-50	0.9693	0.9693	0.9693	0.9693
RF-100	0.9693	0.9693	0.9693	0.9693
RF-200	0.9693	0.9693	0.9693	0.9693
DT	0.9685	0.9684	0.9682	0.9684
KNN	0.9646	0.9646	0.9646	0.9646
Stack classifier (our method)	0.9895	0.9895	0.9895	0.9895

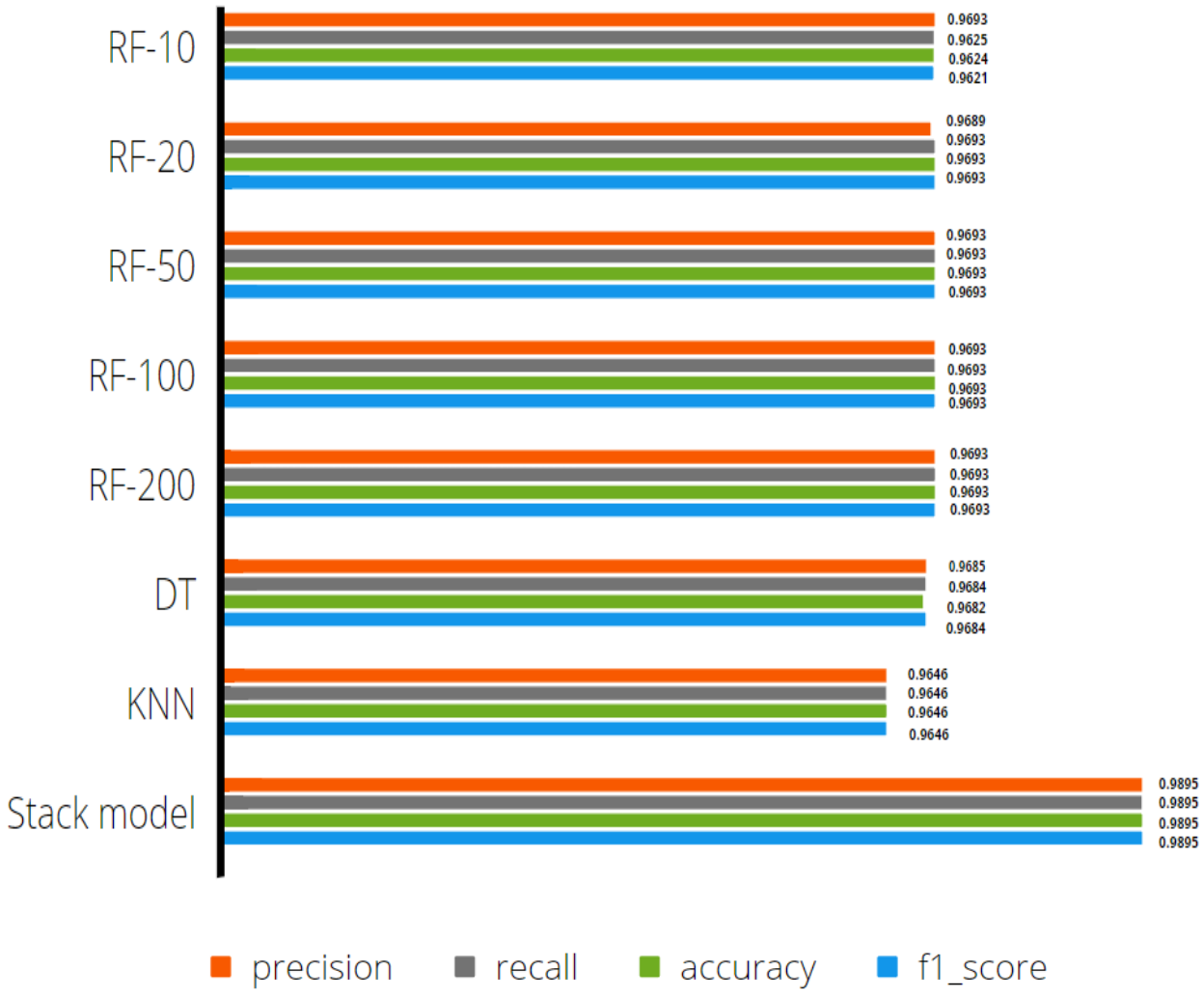

Fig-6: Performance of various classifiers on CICIDS2017 dataset.

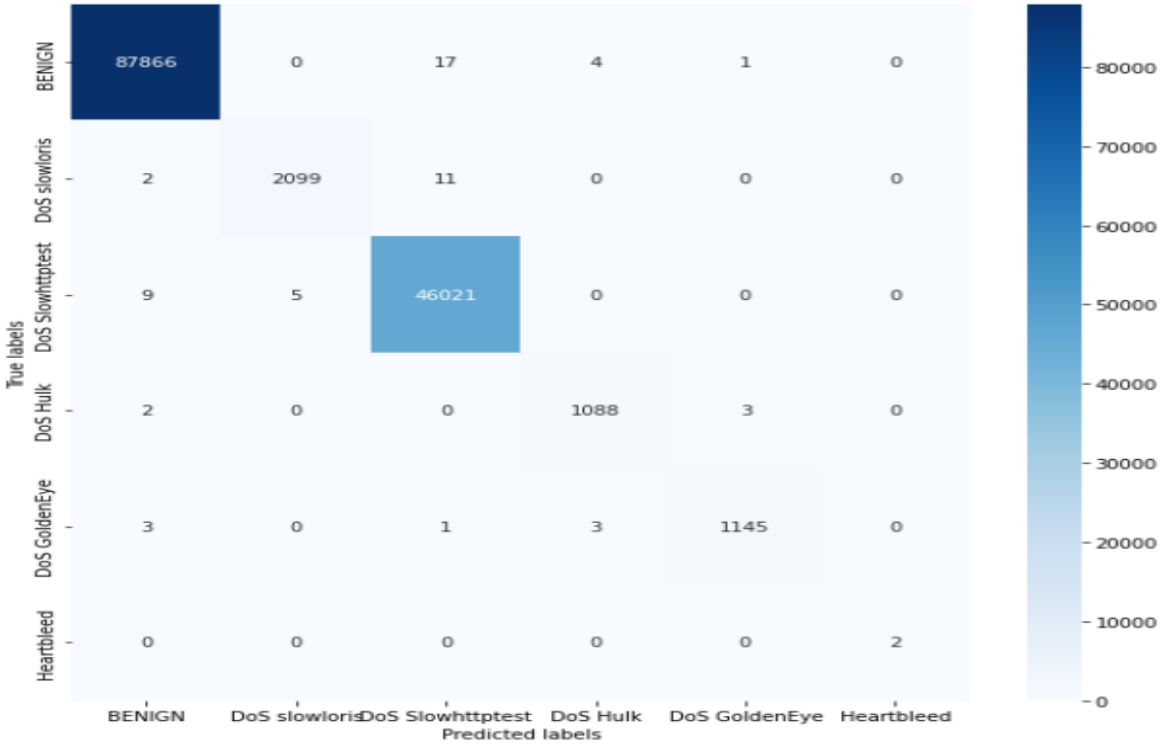

Fig-7: Confusion matrix of multiclass classification of proposed method on cicids2017 dataset.

Table 2: Comparison of results of proposed method and previous related works on CICIDS2017 dataset.

	Reference + method	Accuracy	Precision	Recall	F1_score
	MDAE + LSTM	0.9850	0.9850	0.9950	0.9826
[25]	Naive Bayes	0.2556	0.7652	0.2504	0.1895
	SVM	0.7995	0.7260	0.7960	0.7330
	DNN	0.9236	0.9623	0.9400	0.9526
	MDAE	0.9010	0.9902	0.9026	0.9125
	LSTM	0.9645	0.9680	0.9800	0.9700
	DT + rule based	0.9912		0.9440	
	RF	0.9550		0.9350	
[12]	REP tree	0.9300		0.9165	
	Multilayer perception	0.8515		0.7780	
	Naive Bayes	0.7445		0.8250	
	Jrip	0.9356		0.9352	
	J48	0.9322		0.9200	
[26]	DBN – SVM		0.9755	0.9788	0.9726
	CNN – LSTM	0.9800			
[27]	CNN	0.9822			
	LSTM	0.9623			
[17]	PCA + RF	0.9880	0.9805	0.9835	0.9880
	AE + RF	0.9900			0.9850
	Our model	**0.9895**	**0.9895**	**0.9895**	**0.9895**

Note: our model values are in bold and missing values are not provides in reference.

5. CONCLUSION AND FUTURE WORK

We evaluated the proposed stacking ensemble method on the CICIDS2017 dataset and have achieved an accuracy and other evaluation metrics of 98.95%. Our results outperform many other results that are achieved using other classifiers. It clearly shows that stacked models exhibit better performance compared to other individual algorithms just because of the fact that stacking classifiers stack different algorithms and combine their predictions. In the upcoming time efforts will be made to improve the model even more and try to find out the best classifiers that need to be used in the segments of stacking classifier i.e., base and meta learners. Furthermore, this research will serve as the basis for further research and investigation to enable the development of effective IDSs (using this kind of better classifiers and ensemble techniques) that can be used in a complex network environment.

REFERENCES

[1] Average annualized cost of cyber-attacks on companies in selected countries in 2018*(in million U.S. dollars)*, statistia.
[2] N. Farnaaz and M. A. Jabbar, "Random forest modeling for network intrusion detection system," Procedia Computer Science, vol. 89, pp. 213–217, 2016.

[3] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on SVM with feature augmentation," Knowledge-Based Systems, vol. 136, pp. 130–139, 2017.

[4] P. S. Bhattacharjee, A. K. M. Fujail, and S. A. Begum, "A comparison of intrusion detection by K-means and fuzzy C-means clustering algorithm over the NSL-KDD dataset," in Proceedings of the 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), IEEE, Chennai, India, 2017.

[5] I. M. Akashdeep, I. Manzoor, and N. Kumar, "A feature reduced intrusion detection system using ann classifier," Expert Systems with Applications, vol. 88, pp. 249–257, 2017.

[6] CICIDS-2017: https://www.unb.ca/cic/ datasets/ids-2017.html

[7] R. M. Elbasiony, E. A. Sallam, T. E. Eltobely, and M. M. Fahmy, "A hybrid network intrusion detection framework based on random forests and weighted k-means," Ain Shams Engineering Journal, vol. 4, no. 4, pp. 753–762, 2013.

[8] H. Yao, Q. Wang, L. Wang, P. Zhang, M. Li, and Y. Liu, "An intrusion detection framework based on hybrid multi-level data mining," International Journal of Parallel Programming, vol. 47, no. 4, pp. 740–758, 2019.

[9] E. Min, J. Long, Q. Liu et al., "Su-IDS: a semi-supervised and unsupervised framework for network intrusion detection," in Proceedings of the International Conference on Cloud Computing and Security, pp. 322–334, Springer, Cham, Switzerland, 2018.

[10] M. R. Karbir, R. Onik, and T. Samad, "A network intrusion detection framework based on bayesian network using wrapper approach," International Journal of Computer Applications, vol. 166, no. 4, pp. 975–8887, 2017.

[11] J. Gu, L. Wang, H. Wang, and S. Wang, "A novel approach to intrusion detection using SVM ensemble with feature augmentation," Computers & Security, vol. 86, pp. 53–62, 2019.

[12] A. Ahmim, L. Maglaras, M. A. Ferrag et al., "A novel hierarchical intrusion detection system based on decision tree and rules-based models," in 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 228– 223, IEEE, Santorini, Greece, 2019

[13] V. Kumar, V. Choudhary, V. Sahrawat et al., "Detecting intrusions and attacks in the network traffic using anomaly based techniques," in Proceedings of the 2020 5th International Conference on Communication and Electronics Systems (ICCES), pp. 554–560, IEEE, Coimbatore, India, 2020.

[14] A. S. Qureshi, A. Khan, N. Shamim, and M. H. Durad, "Intrusion detection using deep sparse auto-encoder and selftaught learning," Neural Computing and Applications, vol. 32, no. 8, pp. 3135–3147, 2020.

[15] S. Nathan, T. N. Ngoc, V. D. Phai et al., "A deep learning approach to network intrusion detection," IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41–50, 2018.

[16] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with svm for network intrusion detection," IEEE Access, vol. 6, pp. 52843–52856, 2018.

[17] R. Abdulhammed, H. Musafer, A. Alessa et al., "Features dimensionality reduction approaches for machine learning based network intrusion detection," Electronics, vol. 8, no. 3, 2019.

[18] G. C. Fernandez and S. Xu, "A case study on using deep ´ learning for network intrusion detection," in Proceedings of the MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM), pp. 1–6, IEEE, Norfolk, VA, USA, 2019.

[19] C. Yin, Y. Zhu, S. Liu, J. Fei, and H. Zhang, "Enhancing network intrusion detection classifiers using supervised adversarial training," Fe Journal of Supercomputing, vol. 76, no. 9, pp. 6690–6719, 2020.

[20] P. Madani and N. Vlajic, "Robustness of deep autoencoder in intrusion detection under adversarial contamination," in Proceedings of the 5th Annual Symposium and Bootcamp, pp. 1–8, ACM, New York, NY, USA, 2018.

[21] Chongzhen Zhang, Yanli Chen, YangMeng, Fangming Ruan , Runze Chen, Yidan Li, and Yaru Yang, "A Novel Framework Design of Network Intrusion Detection Based on Machine Learning Techniques", Hindawi Security and Communication Networks Volume 2021, ArticleID 6610675.

[23] Logistic Regression in machine learning, javatpoint

[24] Razan Abdulhammed , Hassan Musafer, Ali Alessa, Miad Faezipour and Abdelshakour Abuzneid" Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection".

[25] H. He, X. Sun, H. He, G. Zhao, L. He, and J. Ren, "A novel multimodal-sequential approach based on multi-view features for network intrusion detection," IEEE Access, vol. 7, pp. 183207–183221, 2019.

[26] H. Zhang, Y. Li, Z. Lv, A. K. Sangaiah, and T. Huang, "A realtime and ubiquitous network attack detection based on deep belief network and support vector machine," IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 3, pp. 790–799, 2020.

[27] P. Sun, P. Liu, Q. Li et al., "DL-IDS: extracting features using CNN-LSTM hybrid network for intrusion detection system," Security and Communication Networks, vol. 2020, Article ID 8890306, 11 pages, 2020.

[28]. Anisha P R , Kishor Kumar Reddy C and Nguyen Gia Nhu, "Blockchain Technology: A Boon at the Pandemic Times – A Solution for Global Economy Upliftment with AI and IoT", EAI/Springer Innovations in Communication and Computing, 2022.

[29]. PR Anisha, CKK Reddy, NG Nhu, Blockchain Technology: A Boon at the Pandemic Times–A Solution for Global Economy Upliftment with AI and IoT, Blockchain Security in Cloud Computing, 227-252, 2022

[30]. PR Anisha, Kishor Kumar Reddy C, NG Nguyen, G Sreelatha, A Text Mining using Web Scraping for Meaningful Insights, Journal of Physics: Conference Series 2089 (1), 012048, 2021

[31]. CKK Reddy, PR Anisha, RM Mohana, Assessing Wear Out of Tyre using Opencv & Convolutional Neural Networks, Journal of Physics: Conference Series 2089 (1), 012001, 2021

[32]. T Lingala, CKK Reddy, BVR Murthy, R Shastry, Y Pragathi, L-Diversity for Data Analysis: Data Swapping with Customized Clustering, Journal of Physics: Conference Series 2089 (1), 012050, 2021

[33]. SN Paidimarry, CKK Reddy, LG Lolla, Internet of things enabled smart baggage follower with theft prevention, Journal of Physics: Conference Series 1950 (1), 012002, 2021

[34]. PR Anisha, CKK Reddy, K Apoorva, CM Mangipudi, Early Diagnosis of Breast Cancer Prediction using Random Forest Classifier, IOP Conference Series: Materials Science and Engineering 1116 (1), 012187, 2021

[35]. RM Mohana, CKK Reddy, PR Anisha, BVR Murthy, Random forest algorithms for the classification of tree-based ensemble, Materials Today: Proceedings, 2021