Indian Journal of Engineering & Materials Sciences
Vol. 29, August 2022, pp. 509-518
DOI: 10.56042/ijems.v29i4.46838

# Machine learning based framework for network intrusion detection system using stacking ensemble technique

Anshu Parashar [*], Kuljot Singh Saggu, & Anupam Garg
Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala 147 001, India

*Received: 25 February 2021; Accepted: 1 September 2021*

Cybersecurity issues are increasing day by day, and it is becoming essential to address them aggressively. An efficient IDS system should be placed to identify abnormal behaviour by dynamically tracing the network traffic pattern. In this work, we proposed a framework for Network Intrusion Detection System using stacking ensemble technique of machine learning, which is testified on Random Forest Regressor and Extra Tree Classifier approaches for feature selections from the subjected dataset. The extensive experimentation has been done by applying 11 states of the art and hybrid machine learning algorithms to select the best performing algorithms. During the investigation, Random Forest, ID3 and XGBoost algorithms are found as best performers among different machine learning algorithms based on accuracy, precision, recall, F1-score and time to increase real-time attack detection performance. Three case studies have been carried out. Our results indicate that the proposed stacking ensemble-based framework of NIDS outperformed compared to the different state of art machine learning algorithms with average 0.99 prediction accuracy.

**Keywords:** Neural Network, Cyber security, Intrusion detection system, Machine learning

## 1 Introduction

The Internet has given distinguished use to the people, which helps them connect. So, it opens a path for illegal access for the people with criminal minds to the Internet's data, which needs to be protected. As the people are unaware of the Internet's vulnerability with the attacks, it leads to sharing of large amounts of data on the Internet, leading to data theft, which is rising with each passing day. To identify such intrusions, a system needs to be deployed that can be known as intrusion detection system (IDS) [1–4]. The Internet is vulnerable to known and unknown attacks. The data on the Internet is secured with the specific designed IDS but cannot protect them from the unknown attacks as the IDS is not designed for them which leave the system exposed to threats. Hence, an IDS should be designed in such a way that it should detect the known and unknown attacks [3,4] The main hazard is for the data related to government facilities that share the sensitive data related to the services provided by them on the cloud storage to contribute to smart cities' vision. Similarly, the automation of medical facilities leads to the necessary amount of patient related data sharing on the Internet leads to illegal use and stealing of data.

A Network Intrusion Detection System (NIDS) monitors the incoming network traffic and responds according to that [1–5]. The NIDS widely focuses on 3 fundamental principles i.e. Confidentiality (the access of the information should lie only in the hands of legitimate user), Integrity (Only legitimate user should be able to modify the information), Availability (The system should be accessible to users always).

On the other hand, the network attacks try to exploit these principles [3,4]. The most common types of network attacks are DoS (Denial of Service attack), Probe (Information gathering), U2R (User to Root) and R2U/R2L (Remote to user/ Remote to local) leads the designers to gain access to the network and send packets from your computer [1–4]. Net Flow Meter (a feature introduced by Cisco in 1996) collected the incoming network traffic, which contains certain features like destination and source IP addresses, Internet control message protocol type, IP protocol, and type of service value. Several ways of intrusion detection methods and the most common practices among those are Signature-based detections and Anomaly-based detection [1–4,6,7]. The system is taught to recognise the normal traffic flow and programmed to flag any anomaly in the traffic flow [6,7]. The IDS process begins with the collection of information from

*Corresponding author (E-mail: aparashar@thapar.edu) various data records or information sources. This data record is a blend of regular and intrusion records, containing the data obtained from log files of networks or host [1–4,6–8]. In the network systems, network anomalies are present which are only be detected by the traditional Network Intrusion Detection Systems. Existing Network Intrusion Detection System are incapable of discovering the latest diversity of attacks present in the network flow. Several machine learning approaches have been investigated and applied to several security related issues. Nowadays, ensemble techniques are finding the focus among the researchers to keep track of the network's malicious activities. Ensemble classifier is an amalgamation of various classifiers that provides more accurate prediction and improves the overall outcome than the single classifier. The cumulative knowledge of the ensemble approaches gives outstanding performance, even if the training data is not sufficient. These properties of ensemble techniques encourage us to frame a model based on stacking ensemble machine learning techniques to improve the performance of the NIDS.

The significant contribution of this work is summarized as follows:

- A framework for Network Intrusion Detection System using stacking ensemble technique using a machine learning approach is proposed in this work.
- For better prediction, the feature selection approaches are applied, namely, Random Forest Regressor and Extra Tree Classifier.
- Eleven state-of-the-art and hybrid machine learning algorithms are applied on the subjected dataset to find best performing classification algorithms and extensive experimentation assessed XGBoost to outstand along with Random Forest and ID3 algorithms.
- The best shortlisted algorithms, namely, Random Forest, ID3 and XGBoost are applied as stacking ensemble approach to increase overall performance.
- Three case studies have been conducted to evaluate the proposed framework based on the accuracy, precision, recall, F1-score and time taken performance metrics.

## 2 Materials and Methods

Several machine learning models have been applied in the field of NIDS. Most of the authors proposed their models based on K-Nearest Neighbor (KNN), Random Forest, Artificial Neural Networks (ANN), Support Vector Machine (SVM), Naive Bayesian (NB) and other basic algorithms. [9–12]. Results of these standalone algorithms were not stable enough, and accuracy was not as expected. In next sub-section, we have discussed brief background includes the related work.

### 2.1 Background

Chitrakar *et al.*[13] applied ISVM on the Kyoto dataset and evaluated the proposed approach in terms of detection rate and error rate. Several studies [14–16] used hybrid techniques of IDS and tried to improve the performance, including feature selection and classification. Essid *et al.* [17] used NB and k2 algorithms to detect an attack. Kulariya *et al.* [18] also applied classification algorithms, namely, logistic regression, support vector machine, random forest, decision trees and Naïve Bayes. If we talk about the feature selection, [19] used correlation-based feature selection and chi-squared feature reduction techniques. Then they have applied different classification techniques, i.e., logistic regression, support vector machine, random forest, etc. on two namely DARPA KDD99 [20] and NSLKDD [21] datasets to detect the attacks. Various works have used hybrid methods for feature selection and classification [22–24]. XGBoost (EXtreme Gradient Boosting) [25] is an advance decision-tree-based ensemble machine learning algorithm. Nowadays, usage of XGBoost is increasing in various fields due to its performance, including speed and scalability[25]. There are limited related works on the use of XGBoost for NIDS. Amaral *et al.* [26] deploy XGBoost to classify malicious traffic in SDNs. Chen *et al.* [27] has applied XGBoost on ACM KDD Cup dataset to train the model, to predict a DDoS. The authors presented that XGboost could detect DDoS attacks by analyzing attack traffic patterns. Chen *et al.* [10] discussed the need for an efficient IDS and used XGBoost to detect an SDN. The experimentations are performed on tcpdump dataset. Xiaolong *et al.* also [28] proposed XGBoost based approach for N-IDS and applied on the KDDCup99 dataset. Their results indicated that XGBoost based NIDS approach giving better accuracy. Further, Dhaliwal *et al.* [29] implemented a similar approach using NSL-KDD dataset, and their results suggested that XGBoost can improve the model's predictability. Bansal *et al.* [30] compared the performance of XGBoost with Naïve Bayes, KNN and AdaBoost algorithms and found XGBoost as the best performer.

In recent years, many researchers have achieved better performance by combining more than one algorithm. Verma *et al.*[5] also proposed ensemble NIDS techniques using XGBoost and AdaBoost models to identify the attacks and avoid false alarms. Pattawaro *et al.* even[31] implemented feature selection, K-Means clustering, and XGBoost classification model for an IDS and experimented on the KDD dataset. Devan and Khare[32] performed the experimentation on NSL-KDD dataset by applying XGBoost and DNN for the classification. The results also compared with logistic regression, SVM, and Naive Bayes models and proposed model outperformed. Bhati *et al.*[33,34] have also shown the usage of XGBoost with ensemble-based IDS, which is helpful for "bias-variance" trade-off. This model is applied to the KDDCup99 dataset. Jiang *et al.* [35] also combined PSO and Xgboost models for the classification and applied on NSL-KDD dataset and compared the proposed model with Xgboost, Random Forest, Bagging and Adaboost. Sharafaldin *et al.* [36] also created CICIDS2017 dataset and further applied different machine learning algorithms for attack prediction. In literature, most of the authors have evaluated their models' performance by computing accuracy, precision, recall, and F1-score metrics[5,33–35,37,38]. In the Network Intrusion Detection System (NIDS), the data stream may get corrupted while transferring across the network, leading to the necessity of applying pre-processing techniques. The pre-processing techniques address the emanates of incorrect, redundant and ambiguous data streams, incorrect data types and the specific feature values which are corrected and replaced in this layer to increase the readability of the features of the data streams so that malicious data can be read to detect the specific data attack [5,33–35,37,38]. After removing the artefacts in the pre-processing layer, data preparation layer brings the specified datasets into the required format and classes depending upon the types of attacks being detected. Further, the model needs to be trained, which is asunder into the training and testing datasets are specified so that the built-up model can be promulgated on the network data stream for the recognition of attacks [5,33–35,37,38].

In case of a NIDS, the importance of feature selection process becomes apparent. It plays a significant and vital role in identifying the pertinent features for the detection of intrusion occurrence. The different feature selection and model building methods are available in the literature [5,33–35,37,38], but the most primarily used method are filtration, wrapper method, embedded methods, boosting, bagging and stacking assembling. An ensemble process is based on the concept of combining multiple models (often called "weak learners") that may produce a more powerful and robust model. This brings us to how these models can be combined, leading to major three types of algorithms that combine these weak learners to accuracy is required. In our model, we have used stacking ensemble [5,33–35,37,38].

The performance of the proposed model is evaluated based on the performance metrics, namely, accuracy, precision, f- measure, recall and execution time defined below [5,33–35,37,38]:

*Accuracy*: The ratio of successfully categorized data to total data.

$$Accuracy = \frac{TN + TP}{FP + TN + TP + FN}$$

*Recall (Sensitivity):* The ratio of data classified as an attack to all attack data.

$$Recall = \frac{TP}{TP + FN}$$

*Precision:* The ratio of successful classified data as the attack to all data classified as the attack.

$$Precision = \frac{TP}{FP + TP}$$

*F-measure (F-score/F1-score):* It represents the harmonic-mean of sensitivity and precision expressing the overall success.

$$F1 - Score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}}$$

**2.2 Framework for machine learning based network IDS using ensemble technique**

The proposed architecture's main objective is to train the model for effective detection of anomalies in the network data streaming using the ensemble of machine learning techniques, as shown in Fig. 1. The proposed framework has been divided into four layers comprises, pre-processing, data preparation, model building and performance evaluation. Before the initialization of the pre-processing phase, our prerequisite step is to understand the dataset used in the proposed work.

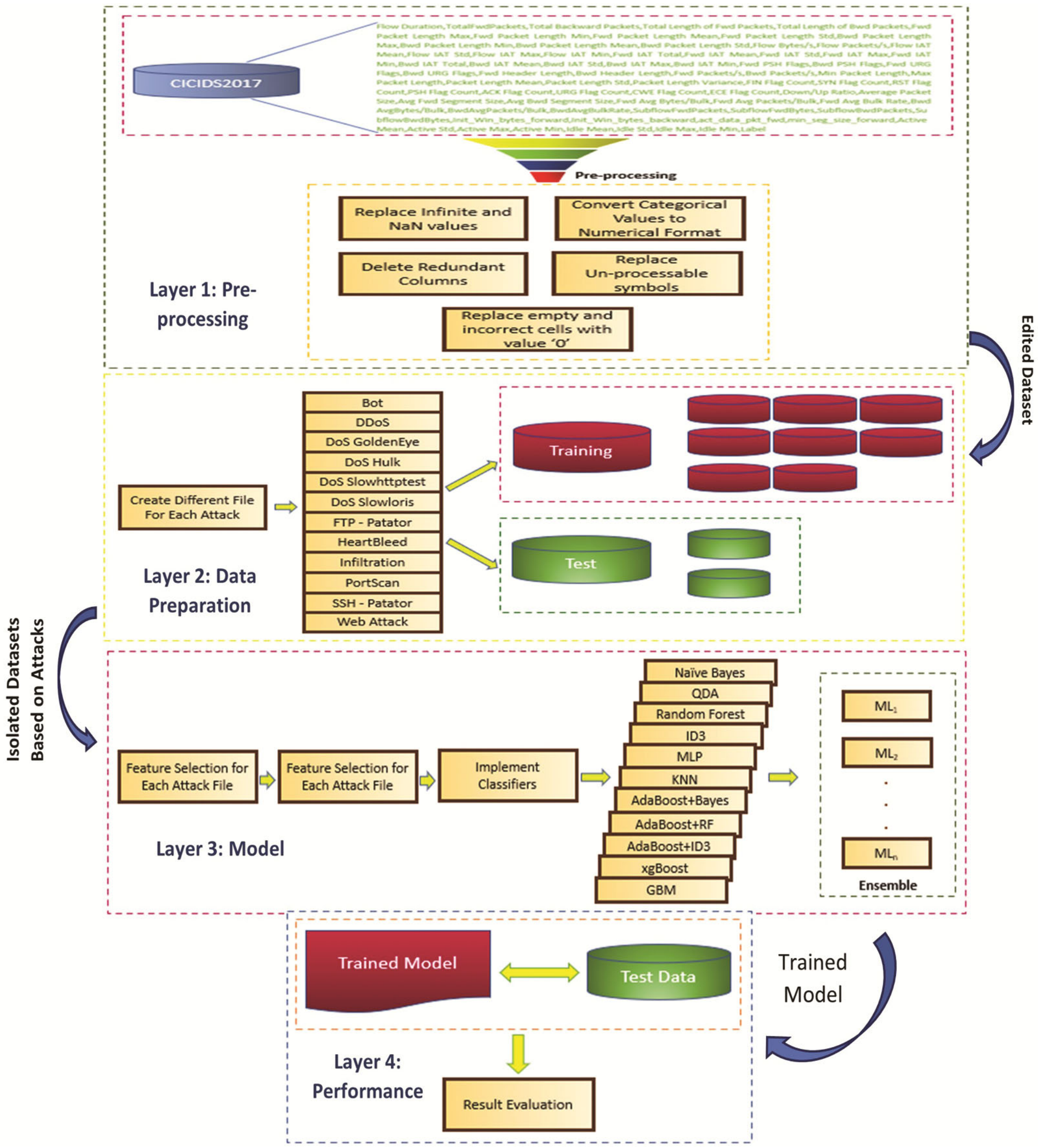
CICIDS2017
Flow Duration,TotalFwdPackets,Total Backward Packets,Total Length of Fwd Packets,Total Length of Bwd Packets,Fwd Packet Length Max,Fwd Packet Length Min,Fwd Packet Length Mean,Fwd Packet Length Std,Bwd Packet Length Max,Bwd Packet Length Min,Bwd Packet Length Mean,Bwd Packet Length Std,Flow Bytes/s,Flow Packets/s,Flow IAT Mean,Flow IAT Std,Flow IAT Max,Flow IAT Min,Fwd IAT Total,Fwd IAT Mean,Fwd IAT Std,Fwd IAT Max,Fwd IAT Min,Bwd IAT Total,Bwd IAT Mean,Bwd IAT Std,Bwd IAT Max,Bwd IAT Min,Fwd PSH Flags,Bwd PSH Flags,Fwd URG Flags,Bwd URG Flags,Fwd Header Length,Bwd Header Length,Fwd Packets/s,Bwd Packets/s,Min Packet Length,Max Packet Length,Packet Length Mean,Packet Length Std,Packet Length Variance,FIN Flag Count,SYN Flag Count,RST Flag Count,PSH Flag Count,ACK Flag Count,URG Flag Count,CWE Flag Count,ECE Flag Count,Down/Up Ratio,Average Packet Size,Avg Fwd Segment Size,Avg Bwd Segment Size,Fwd Avg Bytes/Bulk,Fwd Avg Packets/Bulk,Fwd Avg Bulk Rate,Bwd AvgBytes/Bulk,BwdAvgPackets/Bulk,BwdAvgBulkRate,SubflowFwdPackets,SubflowFwdBytes,SubflowBwdPackets,SubflowBwdBytes,Init_Win_bytes_forward,Init_Win_bytes_backward,act_data_pkt_fwd,min_seg_size_forward,Active Mean,Active Std,Active Max,Active Min,Idle Mean,Idle Std,Idle Max,Idle Min,Label
Pre-processing
Replace Infinite and NaN values
Convert Categorical Values to Numerical Format
Delete Redundant Columns
Replace Un-processable symbols
Replace empty and incorrect cells with value '0'
Layer 1: Pre-processing
Edited Dataset
Create Different File For Each Attack
Bot
DDoS
DoS GoldenEye
DoS Hulk
DoS Slowhttptest
DoS Slowloris
FTP - Patator
HeartBleed
Infiltration
PortScan
SSH - Patator
Web Attack
Training
Test
Layer 2: Data Preparation
Isolated Datasets Based on Attacks
Feature Selection for Each Attack File
Feature Selection for Each Attack File
Implement Classifiers
Naïve Bayes
QDA
Random Forest
ID3
MLP
KNN
AdaBoost+Bayes
AdaBoost+RF
AdaBoost+ID3
xgBoost
GBM
ML1
ML2
MLn
Ensemble
Layer 3: Model
Trained Model
Test Data
Trained Model
Layer 4: Performance
Result Evaluation

Fig. 1 — Proposed framework for network intrusion detection system.

### *2.2.1 Description of the dataset*

In this work, the CICIDS2017 dataset [36] has been used. According to the author of the CICIDS2017, the dataset has over eight different files that contain five days normal and attack related traffic data of Canadian Institute of Cybersecurity. For the experimentation, the defects present in the CICIDS2017 dataset are corrected and edited. Initially, the dataset files have 3119245 of data stream records, it has been observed that there are as many as 288805 data streams that are either incomplete or incorrect which need to be removed.

### 2.2.2 Methodology of the proposed framework

*Layer 1: Pre-processing*

After removing the incorrect data streams, there is a need to eliminate the redundant data streams from the specified dataset. In the raw dataset of CICIDS2017, there are 85 features, few of them namely, Flow ID, Source IP, Source Port, Destination IP, Destination Port and so on. But the feature namely, Fwd. Header Length has its existence twice in the dataset (41st and 62nd column), which leads to the removal of this error by deleting one column. On the analysis of the dataset, it has been detected that there are many features like timestamp, External IP, which have categorical and string values, that are converted to numerical values. Further, all “NaN” and “Infinity” values are replaced with 0 and -1 respectively from the data streams. The Label feature specifies “-“(Unicode &#8211) for defining the web attacks, that cannot be recognised in the implementation language, will be replaced with “- “ (Unicode &#45) so that implementation language recognises this data. The algorithm for this process is given in Table 1.

*Layer 2: Data Preparation*

For the creation of Intrusion Detection System which can detect any type of attack from the specified attacks, these eight files are merged to form a single file which is pre-processed as described earlier. Then, from this file, every attack type of the dataset is isolated from other attacks and a new file is created for it. In this way, we created 12 new files which contain the entire stream identified as the attack with the randomly selected data stream “Benign” (Attack 30%, Benign 70%).

There is a requirement of data for training the machine learning model and test data to evaluate the algorithm's performance. But, CICIDS2017 dataset is not provided with any explicit testing and training data provided. Thus, the data in 12 datasets are divided into training and test data using Sklearn and train_test_split command with 80% training and 20% testing data which becomes the input to the feature selection process. This ratio can also be changed, but 80:20 is preferred so that the model can be trained on extensive data for achieving better accuracy at the time of attack detection. The algorithm of the same is in Table 2.

*Layer 3: Model Building*

CICIDS2017 dataset is a labelled dataset with a total number of 86 features excluding the last column (class label) that specifies the traffic status. These features are extracted through CICFlowMeter-V3(a flow-based extractor) from pcap files with the output in the excel format. The flow label includes SourceIP, SourcePort, DestinationIP, DestinationPort and Protocol that are labelled based on daily attack schedule. Feature selection is done from the dataset using the Random Forest Regressor class of Scikit-learn, defining it as feature set 1. Then, the Extra Tree Classifier class of Scikit learn calculates feature importance and compares its performance with the random forest regressor. Random Forest Regressor creates a decision forest that assigns weightage to the feature based on the importance of the feature in the decision tree's construction. At the end of the process, all these importance weights are compared and sorted. The decision tree's total importance weight is the sum of all the weights given to the features. If the comparison is made between the total importance weight of the decision tree and individual importance, it provides the significance of that feature in the decision tree. Extra Tree Classifier is very similar to random forest Regressor with the only difference in the construction of decision trees in the forest. In extra tree classifier, the creation of forest is done from the training samples. At each test node, a random sample of k features is provided to trees from which

Table 1 — Algorithm 1 for Pre-processing

```
Input: list of 8 csv files from CICIDS2017 dataset
Output: Single csv file formed through combination of input
Begin:
Open file f
For each file i in input list
Replace '-'(Unicode:8211) with '-'(Unicode:45)
Fill empty cells with value '0'
Replace values 'Infinity' and 'NaN' with '-1' and '0' respectively
Convert categorical values to numerical values
Append file i in file f
End for
Close file f
End
```

Table 2 — Algorithm 2 for data preparation

```
Input: Csv file created in pre-processing stage(all_data),list of Number of instances for each attack type(list_n)
Output: 12 csv files categorized based on attack type
Begin:
For i in input list_n
open file f
open input file all_data
write classes form all_data in f
benign_num = (benign/(list_n[i]*(7/3)))
readline from input file all_data and split
if all_data[83] is "BENIGN" then
if number of benign members added less than benign_num
write line in file f
if all_data[83] is i then
write line in file f
close file f
End for
// Web attack files are merged together to create a single file
For each web attack i
read file f of web attack i
Append to new file d
close f
close file d
End for
End
```

each decision tree selects best features to split data typically using the Gini Index. Later, weights are given to the features in the same manner as in Random Forest Regressor. For each decision tree in Extra Tree Classifier or Random Forest Regressor, feature/node importance is calculated using Gini Importance in Scikit-learn. If the assumption is made to have a tree with only two child nodes, then,

$$ni_j = w_j c_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)}$$

$ni_j$ = the importance of node j
$w_j$ = weighted number of samples reaching node j
$c_j$ = impurity value of node j
left(j) = child node from left split on node j
right(j) = child node from right split on node j

The importance is then calculated by:

$$fi_i = \frac{\sum_{j:node\ j\ splits\ on\ feature\ i} ni_j}{\sum_{k\ \in all\ nodes} ni_k}$$

$fi_i$ = importance of feature i
$ni_i$ = importance of node j

Values can be normalized by dividing them with the sum of all feature importance values:

$$normfi_i = \frac{fi_i}{\sum_{j\ \in all\ features} fi_j}$$

The final feature importance is calculated by dividing the sum of the feature's importance value on each tree with the total number of trees:

$$RFfi_i = \frac{\sum_{j\ \in all\ trees} fi_{ij}}{T}$$

$RFfi_i$ = the importance of feature i calculated from all trees in the random forest model
$normfi_{ij}$ = normalized feature importance for i in tree j
T = total number of trees

Any misleading or noncontributing features should be eliminated for improving the efficiency and accuracy in calculating the importance of features. From the specified 85 features in the dataset, elimination of Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp and External IP features is done. The feature such as IP address would be misleading as the attacker can use a fake IP. The Protocol and Timestamp features are non-contributing features that also need to eliminate. The ports for classification can also be misleading as many applications are transmitted using similar ports which would be difficult to differentiate. Hence, any feature related to ports should also be removed. Detail steps are discussed in Table 3.

## 3 Results and Discussion

The important features are extracted to accurately detect each attack after performing the feature selection process on 12 different files. On the top four features of feature set 1 and set 2, eleven different machine learning methods are applied to each attack file ten times, giving a different outcome for each attack type. With this method, we aim to observe the accuracy, effectiveness and performance of machine learning algorithms on all the attack types. After analyzing the results, we build an ensemble classifier with the top three classifiers with maximum accuracy and least processing time. Before ensemble, the top classifiers for different feature selection methods that are Random Forest Regressor and Extra Tree Classifier are compared. The comparison concludes the set of features that provide the highest accuracy for each attack type. That specific feature set can be used for the respective attack type that gives maximum accuracy. This proposed model is the ensemble model.

### 3.1 Case Study 1

In case study 1, the state of the art and hybrid machine learning algorithms mentioned in layer 3 are applied to the features extracted using Random Forest Regressor for the feature selection. The Tables shown below depict the classification algorithms' performance in terms of accuracy, recall, precision, F1-score, and time taken. Table 4 indicates that Random Forest, Decision Tree, K Nearest Neighbor, Adaboost with ID3, Adaboost with Random Forest,

Table 3 — Algorithm 3 for feature selection

```
Input: list of 12 csv files categorized based on attack type (csv_files)
Output: CSV file containing top 4 features of each attack and graph showing the top 20 features
Begin:
For j in csv_files
read csv file
replace values bigger than range of float32 with nan
fil nan values with value 0
If j['Label'] is BENIGN then
Change value of 'Label' column in j with '1'
else
Change value of 'Label' column in j with '0'
Create X and y datafame from j
Apply Random Forest Regressor or Extra Tree Classifier on X,y
Get the results and form graphs
write importance value in another file
End for
End
```

Table 4 — Accuracy performance metric with feature set 1 and feature set 2

Feature Set 1

| Classifier | NB | QDA | RF | ID3 | MLP | KNN | ABNB | ABRF | ABID3 | XGB | GBM |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Attacks | | | | | | | | | | | |
| Bot | 0.54 | 0.65 | 0.95 | 0.95 | 0.68 | 0.95 | 0.66 | 0.97 | 0.97 | 0.97 | 0.97 |
| DDoS | 0.73 | 0.64 | 0.99 | 0.99 | 0.76 | 0.95 | 0.54 | 0.99 | 0.99 | 0.99 | 0.99 |
| Dos GoldenEye | 0.89 | 0.7 | 0.99 | 0.99 | 0.75 | 0.98 | 0.71 | 1 | 1 | 0.99 | 1 |
| DoS Hulk | 0.35 | 0.42 | 0.94 | 0.96 | 0.94 | 0.96 | 0.4 | 0.96 | 0.96 | 0.96 | 0.96 |
| DoS Slowhttptest | 0.4 | 0.45 | 0.98 | 0.98 | 0.83 | 0.99 | 0.68 | 1 | 1 | 0.98 | 1 |
| DoS Slowloris | 0.42 | 0.5 | 0.95 | 0.96 | 0.81 | 0.94 | 0.78 | 0.96 | 0.96 | 0.96 | 0.96 |
| FTP – Patator | 1 | 1 | 1 | 1 | 1 | 1 | 0.22 | 1 | 1 | 1 | 1 |
| HeartBleed | 1 | 1 | 1 | 1 | 0.59 | 1 | 1 | 1 | 1 | 1 | 1 |
| Infiltration | 0.83 | 0.88 | 0.96 | 0.93 | 0.4 | 0.92 | 0.33 | 0.96 | 0.94 | 0.88 | 1 |
| PortScan | 0.44 | 0.83 | 1 | 1 | 0.73 | 1 | 0.56 | 1 | 1 | 1 | 1 |
| SSH - Patator | 0.44 | 0.51 | 0.96 | 0.96 | 0.9 | 0.95 | 0.79 | 0.96 | 0.96 | 0.96 | 0.96 |
| Web Attacks | 0.69 | 0.84 | 0.97 | 0.97 | 0.7 | 0.95 | 0.85 | 0.97 | 0.97 | 0.97 | 0.97 |

Feature Set 2

| Classifier | NB | QDA | RF | ID3 | MLP | KNN | ABNB | ABRF | ABID3 | XGB | GBM |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Attacks | | | | | | | | | | | |
| Bot | 0.74 | 0.93 | 0.94 | 0.93 | 0.93 | 0.94 | 0.71 | 0.94 | 0.94 | 0.93 | 0.94 |
| DDoS | 0.76 | 0.74 | 0.98 | 0.98 | 0.81 | 0.95 | 0.61 | 0.98 | 0.98 | 0.98 | 0.98 |
| Dos GoldenEye | 0.94 | 0.94 | 0.99 | 0.98 | 0.77 | 0.97 | 0.72 | 0.99 | 0.99 | 0.99 | 0.99 |
| DoS Hulk | 0.87 | 0.84 | 0.92 | 0.92 | 0.73 | 0.93 | 0.77 | 0.94 | 0.94 | 0.93 | 0.94 |
| DoS Slowhttptest | 0.88 | 0.91 | 0.97 | 0.97 | 0.82 | 0.99 | 0.41 | 0.99 | 0.99 | 0.97 | 0.99 |
| DoS Slowloris | 0.78 | 0.37 | 0.95 | 0.96 | 0.84 | 0.95 | 0.76 | 0.96 | 0.96 | 0.96 | 0.97 |
| FTP - Patator | 0.39 | 0.84 | 0.93 | 0.93 | 0.88 | 0.93 | 0.79 | 0.94 | 0.94 | 0.93 | 0.94 |
| HeartBleed | 1 | 1 | 1 | 1 | 0.78 | 1 | 1 | 1 | 1 | 1 | 1 |
| Infiltration | 0.88 | 0.88 | 0.91 | 0.96 | 0.52 | 0.92 | 0.28 | 0.92 | 0.94 | 1 | 0.88 |
| PortScan | 0.94 | 0.94 | 0.98 | 0.98 | 0.98 | 0.98 | 0.71 | 0.98 | 0.98 | 0.98 | 0.98 |
| SSH - Patator | 0.39 | 0.7 | 0.96 | 0.96 | 0.87 | 0.96 | 0.82 | 0.96 | 0.96 | 0.96 | 0.96 |
| Web Attacks | 0.91 | 0.46 | 0.96 | 0.96 | 0.84 | 0.97 | 0.34 | 0.96 | 0.96 | 0.96 | 0.96 |

NB= Naïve Bayes, QDA= Quadratic Discriminant Analysis, RF= Random Forest, ID3= Iterative Dichotomiser 3, MLP= Multi-Level Perceptron, KNN=k-Nearest Neighbor, ABNB=AdaBoost with Naïve Bayes, ABRF= AdaBoost with Random Forest, ABID3=AdaBoost with ID3, XGB=Extreme Boosting, GBM= Gradient Boosting

XgBoost and GBM have achieved around 95% accuracy. Similarly, in Table 5, f-1 score is almost equal to 1 for every attack. Gradient Boosting has achieved the highest accuracy in 11 out of 12 cases among the specified classification algorithms. As per the expectations, GBM classifier considers the huge amount of time for detecting attacks that is not appropriate for the model's performance. Moreover, kNN classification algorithm outperforms in accuracy, but it consumes maximum computation time. The classification algorithms that stand out with the highest accuracy and low processing time from the implemented classification techniques are Random Forest, ID3 and XgBoost. Moreover, Naive Bayes and QDA are the algorithms that have the lowest score in most of the attacks detection. But it is worth mentioning that Naive Bayes took the least time in every detection. Interestingly, Heartbleed and FTP-Patator attack noted that almost all the classification approaches had achieved 99% accuracy. This could be because the number of data streams for both attacks is less, so it is much easier to distinguish features that are easily able to determine if there is a Heartbleed or FTP-Patator attack or not.

### 3.2 Case Study 2

In case study 2, the existing and hybrid machine learning algorithms mentioned in layer 3 are applied to the features extracted using Extra Tree Classifier for the feature selection. The Table 6 shows below depict the classification algorithms' performance in terms of accuracy, recall, precision, F1-score, and time taken. It has been observed from the values of Table 4, Table 5 and Table 6 that the results for ID3, RF, Xg Boost and GBM are showing comparatively similar performance as with the features extracted using Random Forest Regressor as described in case study 1. But it is worth noticing that the accuracy and precision for Naïve Bayes, Quadratic Discriminant Analysis, MLP and Adaboost with Naïve Bayes have significantly increased. As in the case of Bot attack, for the feature set 1, we have 54% accuracy for Naïve

Table 5 — F1 Score performance metric for feature set 1 and feature set 2

Feature Set 1

| Classifier / Attacks | NB | QDA | RF | ID3 | MLP | KNN | ABNB | ABRF | ABID3 | XGB | GBM |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bot | 0.53 | 0.65 | 0.94 | 0.95 | 0.58 | 0.94 | 0.4 | 0.97 | 0.97 | 0.96 | 0.97 |
| DDoS | 0.73 | 0.51 | 0.99 | 0.99 | 0.73 | 0.95 | 0.51 | 0.99 | 0.99 | 0.99 | 0.99 |
| Dos GoldenEye | 0.86 | 0.7 | 0.99 | 0.99 | 0.7 | 0.97 | 0.45 | 1 | 1 | 0.99 | 1 |
| DoS Hulk | 0.31 | 0.4 | 0.92 | 0.95 | 0.93 | 0.95 | 0.40 | 0.96 | 0.96 | 0.95 | 0.96 |
| DoS Slowhttptest | 0.38 | 0.44 | 0.98 | 0.98 | 0.8 | 0.98 | 0.45 | 1 | 1 | 0.98 | 0.99 |
| DoS Slowloris | 0.4 | 0.5 | 0.94 | 0.95 | 0.8 | 0.93 | 0.7 | 0.96 | 0.96 | 0.95 | 0.95 |
| FTP - Patator | 1 | 1 | 1 | 1 | 1 | 1 | 0.18 | 1 | 1 | 1 | 1 |
| HeartBleed | 1 | 1 | 1 | 1 | 0.53 | 1 | 1 | 1 | 1 | 1 | 1 |
| Infiltration | 0.75 | 0.84 | 0.95 | 0.92 | 0.37 | 0.89 | 0.29 | 0.95 | 0.95 | 0.85 | 1 |
| PortScan | 0.43 | 0.82 | 1 | 1 | 0.52 | 1 | 0.44 | 1 | 1 | 1 | 1 |
| SSH - Patator | 0.43 | 0.5 | 0.95 | 0.95 | 0.88 | 0.94 | 0.72 | 0.96 | 0.96 | 0.95 | 0.96 |
| Web Attacks | 0.69 | 0.83 | 0.96 | 0.96 | 0.63 | 0.94 | 0.83 | 0.97 | 0.97 | 0.96 | 0.96 |

Feature Set 2

| Classifier / Attacks | NB | QDA | RF | ID3 | MLP | KNN | ABNB | ABRF | ABID3 | XGB | GBM |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Bot | 0.74 | 0.92 | 0.93 | 0.92 | 0.93 | 0.93 | 0.41 | 0.93 | 0.93 | 0.92 | 0.93 |
| DDoS | 0.76 | 0.74 | 0.98 | 0.98 | 0.78 | 0.95 | 0.51 | 0.98 | 0.98 | 0.98 | 0.98 |
| Dos GoldenEye | 0.92 | 0.93 | 0.98 | 0.98 | 0.73 | 0.96 | 0.60 | 0.99 | 0.99 | 0.99 | 0.99 |
| DoS Hulk | 0.89 | 0.80 | 0.90 | 0.91 | 0.54 | 0.92 | 0.73 | 0.93 | 0.93 | 0.92 | 0.93 |
| DoS Slowhttptest | 0.86 | 0.89 | 0.96 | 0.96 | 0.78 | 0.99 | 0.4 | 0.99 | 0.99 | 0.96 | 0.99 |
| DoS Slowloris | 0.72 | 0.33 | 0.94 | 0.96 | 0.82 | 0.95 | 0.65 | 0.96 | 0.96 | 0.95 | 0.96 |
| FTP - Patator | 0.36 | 0.83 | 0.92 | 0.91 | 0.85 | 0.92 | 0.72 | 0.93 | 0.93 | 0.91 | 0.93 |
| HeartBleed | 1 | 1 | 1 | 1 | 0.75 | 1 | 1 | 1 | 1 | 1 | 1 |
| Infiltration | 0.86 | 0.84 | 0.89 | 0.95 | 0.45 | 0.90 | 0.25 | 0.9 | 0.92 | 1 | 0.86 |
| PortScan | 0.93 | 0.93 | 0.98 | 0.98 | 0.98 | 0.98 | 0.65 | 0.98 | 0.98 | 0.98 | 0.98 |
| SSH - Patator | 0.36 | 0.69 | 0.96 | 0.95 | 0.86 | 0.95 | 0.76 | 0.96 | 0.96 | 0.96 | 0.96 |
| Web Attacks | 0.90 | 0.45 | 0.96 | 0.95 | 0.76 | 0.96 | 0.33 | 0.96 | 0.96 | 0.96 | 0.96 |

Table 6 — Comparison of various performance metrics using feature set 1 and feature set 2

| Criteria | Average Accuracy | | Average Recall | | Average Precision | | Average F1 Score | |
|---|---|---|---|---|---|---|---|---|
| Classifiers | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 |
| Naïve Bayes | 0.63 | 0.79 | 0.72 | 0.81 | 0.77 | 0.83 | 0.63 | 0.78 |
| QDA | 0.68 | 0.80 | 0.76 | 0.82 | 0.79 | 0.83 | 0.68 | 0.78 |
| RF | 0.97 | 0.96 | 0.97 | 0.95 | 0.97 | 0.95 | 0.97 | 0.95 |
| ID3 | 0.97 | 0.96 | 0.97 | 0.96 | 0.97 | 0.96 | 0.97 | 0.95 |
| MLP | 0.70 | 0.81 | 0.75 | 0.80 | 0.76 | 0.75 | 0.70 | 0.77 |
| KNN | 0.96 | 0.96 | 0.96 | 0.95 | 0.96 | 0.95 | 0.96 | 0.95 |
| ABNB | 0.53 | 0.66 | 0.58 | 0.63 | 0.63 | 0.64 | 0.53 | 0.58 |
| ABRF | 0.98 | 0.96 | 0.98 | 0.96 | 0.98 | 0.96 | 0.98 | 0.96 |
| ABID3 | 0.97 | 0.97 | 0.97 | 0.96 | 0.98 | 0.96 | 0.97 | 0.96 |
| XGB | 0.97 | 0.97 | 0.97 | 0.96 | 0.96 | 0.96 | 0.97 | 0.96 |
| GBM | 0.98 | 0.96 | 0.98 | 0.96 | 0.98 | 0.95 | 0.98 | 0.96 |

Bayes, but with the feature set 2, the accuracy boosted to 76%. Similarly, the case with Quadratic Discriminant Analysis, Multi-Level Perceptron and Adaboost with Naïve Bayes classification techniques, where the accuracy has increased significantly in most cases.

In the case of study 2 performance evaluation, it can be observed that for NB, QDA, MLP and ADNB, the model's performance increased. It is discovered that for Naïve Bayes, QDA, MLP and AdaBoost with Naïve Bayes classifiers, there is a significant increase in performance of the model on the usage of feature set 2 as compared to feature set 1. Whereas for the remaining classification techniques, usage of feature set 1 by the model gives the better performance. The performance metric time taken for algorithms is almost same for both feature sets, so, time taken is not used for comparison.

### 3.3 Case Study 3-Stacking Ensemble

Random Forest, ID3 and XgBoost outperforms compared to other classification techniques, leading us to make a stacking ensemble approach with them in order to increase the prediction accuracy of the

Table 7 — Result for Ensemble technique with features of Feature set 1 and Feature set 2

| Criteria | Accuracy | | Recall | | Precision | | F1-Score | | Time | |
|---|---|---|---|---|---|---|---|---|---|---|
| Attacks | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 | Feature Set 1 | Feature Set 2 |
| Bot | 0.97 | 0.94 | 0.97 | 0.95 | 0.95 | 0.91 | 0.96 | 0.93 | 0.57 | 0.49 |
| DDoS | 0.99 | 0.98 | 0.99 | 0.98 | 0.99 | 0.98 | 0.99 | 0.98 | 11.18 | 10.52 |
| Dos GoldenEye | 0.99 | 0.99 | 0.99 | 0.98 | 0.99 | 0.99 | 0.99 | 0.99 | 1.67 | 1.76 |
| DoS Hulk | 0.96 | 0.93 | 0.95 | 0.92 | 0.95 | 0.92 | 0.95 | 0.92 | 32.24 | 37.45 |
| DoS Slowhttptest | 0.99 | 0.97 | 0.99 | 0.96 | 0.99 | 0.97 | 0.99 | 0.97 | 1.02 | 1.21 |
| DoS Slowloris | 0.97 | 0.96 | 0.95 | 0.95 | 0.97 | 0.97 | 0.96 | 0.96 | 1.07 | 1.23 |
| FTP- Patator | 1 | 0.93 | 1 | 0.92 | 1 | 0.92 | 1 | 0.92 | 1.04 | 1.26 |
| HeartBleed | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.33 | 0.33 |
| Infiltration | 0.98 | 0.96 | 0.96 | 0.93 | 0.99 | 0.97 | 0.97 | 0.95 | 0.33 | 0.35 |
| PortScan | 1 | 0.98 | 1 | 0.99 | 1 | 0.97 | 1 | 0.98 | 23.82 | 17.14 |
| SSH – Patator | 0.96 | 0.96 | 0.96 | 0.97 | 0.94 | 0.95 | 0.95 | 0.96 | 1.17 | 1.13 |
| Web Attacks | 0.96 | 0.96 | 0.94 | 0.94 | 0.97 | 0.97 | 0.96 | 0.95 | 0.66 | 0.73 |

Table 8 — Comparison with the state-of-art related work

| Classifiers | Sharafaldin *et al.* [8] | | | | Proposed Study | | | |
|---|---|---|---|---|---|---|---|---|
| | Accuracy | Precision | Recall | F-1 Score | Accuracy | Precision | Recall | F-1 Score |
| NB | - | 0.88 | 0.84 | 0.84 | 0.85 | 0.88 | 0.85 | 0.84 |
| QDA | - | 0.97 | 0.88 | 0.92 | 0.85 | 0.86 | 0.87 | 0.83 |
| RF | - | 0.98 | 0.97 | 0.97 | 0.67 | 0.98 | 0.98 | 0.98 |
| ID3 | - | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 |
| MLP | - | 0.77 | 0.83 | 0.76 | 0.84 | 0.85 | 0.85 | 0.83 |
| KNN | - | 0.96 | 0.96 | 0.96 | 0.97 | 0.97 | 0.98 | 0.98 |
| ABNB | - | - | - | - | 0.73 | 0.73 | 0.68 | 0.64 |
| ABRF | - | - | - | - | 0.98 | 0.99 | 0.99 | 0.99 |
| ABID3 | - | 0.77 | 0.84 | 0.77 | 0.98 | 0.99 | 0.99 | 0.99 |
| XGB | - | - | - | - | 0.98 | 0.99 | 0.99 | 0.99 |
| GBM | - | - | - | - | 0.99 | 0.99 | 0.99 | 1 |
| Proposed Model | - | - | - | - | 0.99 | 0.98 | 0.98 | 0.98 |

model. It has been observed from Table 7 that proposed ensemble model gives accuracy around 96% and nearly 100% specifically for FTP-Patator, Heart Bleed and Port Scan attacks. The same is also observed in recall and precision. The feature set 1 has better recall and precision than the feature set 2. F1 scores are better for the feature set 1 except in attack class SSH – Patator, where f-score for the feature set 1, is 0.95 whereas for the feature set 2 it is 0.96. Moreover, it is noticed that, feature set 2 takes less time in processing than features set 1, but there is not a significant difference between there processing time. Overall, the predictive performance for the feature set 1 is better as compared to feature set 2. The proposed study's experimentation results are compared with Sharafaldin *et al.* [36] (creator of CICIDS 2017 dataset) in the Table 8.

Table 8 shows us the comparison of results between the two studies that are highlighting the maximum values. Accuracy was not used as a parameter in Sharafaldin *et al.* [36] for evaluation. Moreover, we used more variety of machine learning algorithms than them. In the case of QDA, it is noticed that Sharafaldin *et al.* [36] model performs better. There is a difference of 0.12, 0.01, and 0.09 points respectively for precision, recall and F1 score. On the other hand, our model performs better for RF, MLP, KNN, AdaBoost. There is a significant difference of 0.22, 0.15, and 0.22 respectively for precision, recall and f1 score for Adaboost with ID3. MLP performs the worst in our study, whereas in our study, whereas Naïve Bayes performs the worst in other studies. GBM performs the best in our study, whereas KNN performs the best in other studies. Next, the final ensemble model is also compared in which Random Forest, ID3 and XG Boost are used to increasing our real-time detection performance with the average accuracy of 0.99.

## 4 Conclusion

Due to the increased involvement of today's generation on the social media and more vulnerable to the technological aspects, the rate for cyber-attacks has grown exponentially, so it needs the hour to control this vulnerability. It leads to the demand for the deployment of an effective IDS to prevent crucial and sensitive data transmission over the Internet. Several IDS systems have been developed by the researcher's, but there is still a room of improvement as they cannot handle the dynamic nature of the attacks. To cover the dynamic nature aspect of the attacks, the use of machine learning algorithms come into the picture, so that the IDS system can train and detect the attacks in the real time. The various machine learning algorithms are explored and become the base foundation for this work. The experimentation in this research work represents the best features which can be used to detect the attacks in the CICIDS2017 dataset using Random Forest Regressor and Extra Tree Classifier techniques. Based on these feature selection approaches, 2 case studies are presented to explore the impact on the detection rate by using the selected features and various machine learning algorithms resulting in the best feature set and best performed classifiers based on the performance metrics, namely, accuracy, precision, recall, F1-score and time taken for the detection. It is concluded that the XGBoost, Random forest and ID3 outperform as compared to the other machine learning algorithms, viz., Naïve Bayes, Quadratic Discriminant Analysis, Multi-Level Perceptron, K- Nearest Neighbours, Ada Boost and Naïve Bayes, Ada Boost and Random Forest, Ada Boost and ID3 and Gradient Boosting. The best performed classifiers are ensemble using a stacking ensemble technique to increase the prediction rate for the dynamic nature of attacks, giving an average accuracy of 99%. It is found out that the performance of feature set 1 is better as compared to the feature set 2, which is selected using the Random Forest Regressor approach. The proposed framework's primary advantage is selecting outperformed features and machine learning algorithms to the ensemble for an efficient prediction that can be deployed to the real-time environment. In future, such a framework may be extended in IoT based smart cyber networks.

## References

1 Lunt T F, *Comput Secur*, 12(1993)405.

2 Hoque M S, Mukit M A, & Bikas M A N, *ArXiv*, (2012)1204.

3 Tarter A, *Spr Int Publ*, (2017)213.

4 Li J, Qu Y, Chao F, Shum H P H, Ho E S L, & Yang L, *Spr Int Publ*, (2019)151.

5 Verma P, Anwar S, Khan S, & Mane S B, in *Int Conf Comp, Comp Net Techno (ICCCNT)*, (2018)1.

6 Buczak A L, & Guven E, *IEEE Commun Surv & Tutorials*, 18(2016)1153.

7 Vasilomanolakis E, Karuppayah S, Mühlhäuser M, & Fischer M, *ACM Comput Surv*, 47(2015)1.

8 Ahmad Z, Khan A S, Shiang C W, Abdullah J, & Ahmad F, *Trans Emerg Telecommun Technol* (2020)1.

9 Mukherjee S, & Sharma N, *Procedia Technol*, 4 (2012)119.

10 Chen F, Ye Z, Wang C, Yan L, & Wang R, in *IEEE Int Symp Wirel Syst within Int Conf Intell Data Acquis Adv Comput Syst,* 68(2018)72.

11 Shenfield A, Day D, & Ayesh A, *ICT Express*, 4 (2018)95.

12 Gu J, Wang L, Wang H, & Wang S, *Comput Secur,* 86(2019)53.

13 Chitrakar R, & Huang C *Comput Secur,* 45(2014)231.

14 Kuang F, Xu W, & Zhang S, *Appl Soft Comput,* 18 (2014)178.

15 Gouveia A, & Correia M, in *IEEE Int Symp Net Comp App (NCA)*, (2016)68.

16 Aljawarneh S A, Aldwairi M, & Yassein M O B, *J Comput Sci*, 25(2018)152.

17 Essid M, & Jemili F, *IEEE Int Conf Syst Man, Cybern*, (2016)4724.

18 Kulariya M, Saraf P, Ranjan R, & Gupta G P, *Int Conf Commun Signal Process*, (2016)1973.

19 Gupta G P, & Kulariya M in *Procedia Comput Sci*, 93(2016)824.

20 University of California, *Information and Irvine Computer Science University of California: KDD Cup Data.*, (1999). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

21 Canadian Institute of Cybersecurity. *UNB-ISCX: NSL KDD Dataset*, (2009). https://www.unb.ca/cic/datasets/nsl.html

22 Kato K, & Klyuev V, in *IEEE Conf Dependable Secur Comput*, (2017)416.

23 Siddique K, Akhtar Z, Lee H, Kim W, & Kim Y, *Symmetry (Basel)*, 9 (2017)197.

24 Soheily-Khah S, Marteau P F, & Béchet N, (2017).

25 Chen T, & Guestrin C, in *Proc ACM SIGKDD Int Conf Knowl Disc Data Min*, (2016)785.

26 Amaral P, Dinis J, Pinto P, Bernardo L, Tavares J, & Mamede H S, in *IEEE 24th Int Conf Net Prot (ICNP),* (2016)1.

27 Chen Z, Jiang F, Cheng Y, Gu X, Liu W, & Peng J, in *IEEE Int Conf Big Data Smart Comput*, (2018)251.

28 Xiaolong X, Wen C, & Yanfei S, *J Syst Eng Electron*, 30(2019)1182.

29 Dhaliwal S S, Nahid A, & Abbas R, *Inf*, 9(2018)149.

30 Bansal A K, & Kaur S, in *Int Conf Adv Compu and Data Sci*, (2018)372.

31 Pattawaro A, & Polprasert C, in *Int Conf ICT Knowl Eng*, (2018)1.

32 Devan P, & Khare N, *Neural Comput Appl*, 32(2020) 12499.

33 Bhati B S, Chugh G, Al-turjman F, & Bhati N S, *Trans Emerg Telecommun Technol*, 32(2021)1.

34 Bhattacharya S, Kaluri S, Singh S, Alazab M, & Tariq U, *Electronics*, 9(2020)219.

35 Jiang H, He Z, Ye G, & Zhang H, *IEEE Access*, 8 (2020)58392.

36 Sharafaldin I, Gharib A, Lashkari A H, & Ghorbani A A, *Soft. Netw*, 1(2017)177.

37 Shen Y, Zheng K, Wu C, Zhang M, Niu X, & Yang Y, *Comput J*, 61(2018)526.

38 Thangaraj M, *Int J Appl Inf Syst*, 5(2013)1.