STM JOURNALS

Current Trends in
Information Technology

ISSN: 2249-4707 (Online)
ISSN: 2348-7895 (Print)
Volume 13, Issue 1, 2023
DOI (Journal): 10.37591/CTIT

http://computerjournals.stmjournals.in/index.php/CTIT/index

Research CTIT

Evaluation of Ensemble and Deep Learning Classifiers on CSE-CIC-IDS2018 Dataset for Intelligent NIDS

Kaushik Datta[1],*, Tapas Samanta[2], Sarbajit Pal[3]

Abstract

Network Intrusion Detection System (NIDS) plays an active role in preventing cyber attacks by early detection of threats before it really starts affecting targeted information services. Over the years many intrusion detection system (IDS) have been developed applying signature or rule-based approach to prevent unauthorised access of network or computer devices. However, ever growing landscape of cyber attacks in recent years has motivated present day researchers to design and develop more accurate IDS using modern Machine Learning (ML) methods which identify attacks through anomaly detection. Development of intelligent NIDS highly depends on a rich, up-to-date and contemporary dataset which consists of relevant attributes and real-world scenario of cyber attacks. Varity of datasets are available for this purpose among which KDDCUP99, NSLKDD, ISCX2012, CICIDS2017, CICIDS2018, Kyoto etc. are the most popular ones and widely used. This paper reports our observations on the performance of two well known classifiers among Ensemble Learning methods, namely Random Forest and XGBoost and of Deep Neural Network classifier on the CSE-CIC-IDS2018 dataset which is relatively a new one and covers many contemporary cyber attacks. Their performances are evaluated using multiple metrics including Precision-Recall curve which has been proved to be more useful in case of imbalanced dataset like CSE-CIC-IDS2018.

Keywords: Network Intrusion Detection System, CSE-CIC-IDS2018 dataset, Ensemble Learning, Multilayer Perceptron, Random Forest, XGBoost, Deep Neural Network.

INTRODUCTION

With the rapid integration of ICT (Information & Communication Technology) in all aspects of human lives, valuable information are generated at larger pace in every corner of the cyber world. At the same time, malicious attempts of breaching privacy, gaining unauthorised access, damaging important data, blocking essential services are also growing faster than ever and becoming more sophisticated and diverse in nature. Hence, cyber attack happens to be a matter of serious concern for the system administrators to safeguard the interest of its users in the cyber space. Network Intrusion Detection Systems (NIDS) plays an active role in preventing such cyber attacks by early detection of threats before it really starts affecting targeted systems. NIDS is not a new concept, rather it has been proposed since the earliest network attacks. However, through last couple of decades, endeavours are also being made to apply machine learning techniques into cyber security domain in order to effectively protect the digital world from newer kinds of threat using anomaly detection based approach instead of traditional signature matching/rule based static methods. Ensemble learning is a kind of supervised machine learning methods which combines the predictive power of

***Author for Correspondence**
Kaushik Datta
E-mail: kaushikdatta@vecc.gov.in

[1]Variable Energy Cyclotron Centre, Department of Atomic Energy, Government of India

Received Date: February 28, 2023
Accepted Date: April 06, 2023
Published Date:

Citation: Kaushik Datta, Tapas Samanta, Sarbajit Pal. Evaluation of Ensemble and Deep Learning Classifiers on CSE-CIC-IDS2018 Dataset for Intelligent NIDS. Current Trends in Information Technology. 2023; 13(1): 1–11p.

several base learners (constituents of the ensemble) to improve the overall predictive ability of the model. Ensemble learning methods such as Random Forest, AdaBoost, XGBoost etc. have gained much popularity among data researchers and practitioners owing to their overwhelming performance specially for structured problems that use tabular datasets. On the other hand, Deep learning is an advanced machine learning approach in which multilayer artificial neural networks are trained to recognize the complex non-linear relation between input and output. Deep learning has successfully demonstrated its incredible performance on a range of problems with unstructured data such as images, video, audio, and text. However, this approach can also be utilized for applications dealing with structured datasets which are large enough and hence, potentially capable to successfully train the deep learning model. In this paper, we have reported our experimental observations on multi-class classification of CSE-CIC-IDS2018 dataset [1] using three methods namely Random Forest, XGBoost and Deep Neural Network (Multilayer Perceptron) where non uniform weights have been incorporated with individual class for ensemble learning to reduce the impact of huge class imbalance and comparative performance of the said methods are evaluated using multiple metrics including *Precision-Recall (PR)* curve. It has been also demonstrated in this paper that the PR curve is more appropriate evaluation metrics than *Receiver Operating Characteristic (ROC)* curve in case of highly imbalanced dataset like CSE-CIC-IDS2018.

The remaining part of this paper is arranged as follows. Section-II gives a brief account of survey conducted on related works with CSE-CIC-IDS2018 dataset. An introduction of the CSE-CIC-IDS2018 dataset has been given in the Section-III. Section-IV details the pre-processing tasks performed on the dataset before it was put into model training. Various classification techniques and Performance Metrics are briefly described in Section-V and Section-VI respectively. Section-VII analyses our experimental observations. Finally, Section-VIII draws the conclusion and future activities.

RELATED WORKS ON CSE-CIC-IDS2018 DATASET

A lot of datasets are available for research on cyber security issues related to intrusion detection - a survey of which is presented in [2] explaining generic features of dataset along with their specific description and comparison. In [3], authors have conducted binary classification (Benign & Attack) on the reduced feature set (23 out of 80) of CSE-CIC-IDS2018 by applying ensemble model which uses voting method to aggregate the individual results of three top performing single classifiers among seven. Karatas et al., in [4], has reported improved performance of six type of learners by addressing class imbalance using Synthetic Minority Oversampling Technique (SMOTE) algorithm [5] while experimenting with a subset (50,00,000 samples out of ~160,00,000) of CSE-CIC-IDS2018 dataset. The LightGBM [6] classification technique has been utilised and evaluated in comparison with other methods namely SVM, RF, Adaboost, MLP, CNN and Naive Bayes in [7] where author has applied an under-sampling and embedded feature selection in order to reduce the class imbalance in CSE-CIC-IDS2018 dataset. Description and results of binary and multi-level classification with Multilayer Perceptron (MLP) [8] on CSE-CIC-IDS2018 dataset using various deep learning frameworks (Keras, TensorFlow, Theano, fast.ai, and PyTorch) have been reported in [9] where authors observed that fast.ai outperforms others. Filho et al [10] has proposed an online approach for DoS/DDoS attack detection using machine learning techniques where a customised dataset along with four well-known ones i.e CIC-DoS, ISCX2012, CICIDS2017 and CICIDS2018 have used for generating random traffic samples and detection is performed on six classifiers i.e Random Forest, Decision Trees, Logistic Regression, SGDClassifier, AdaBoost, MLP. A method based on Convolution Neural Network (CNN) [11] – a widely used deep learning method for image recognition, has been proposed in [12] for detection of DoS attack from KDDCup 1999 and CSE-CIC-IDS2018 datasets by converting all numerical samples into image data. The authors in [13] has run a two-layer MLP on a sample of 10, 48,575 records of CSE-CIC-IDS2018 to detect only bot attacks and used GridSearchCV techniques for hyper-parameter optimization.

DESCRIPTION OF CSE-CIC-IDS2018 DATASET

Realistic and representative network dataset is a common and essential requirement for any ML-based approach to design a successful network intrusion detection system. However, the lack of such realistic and publicly available datasets which is truly representational and optimal in terms of benign and anomalous network traffic poses biggest challenges in the advancement of designing anomaly based NIDS [14]. Moreover, due to the rapid evolution of intrusion pattern and change of network behaviour, researchers and developers are always in hunt of newer datasets which embodies contemporary traffic compositions and attack scenario.

CSE-CIC-IDS2018 dataset which is available on AWS (Amazon Web Service) cloud is an outcome of a collaborative project between the Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC) [1].

CSE-CIC-IDS2018 dataset (Table 1) was created over a period of 10 days within an emulated environment as described in [1]. This dataset contains raw data in the form of network packet (*.pcap* file) and event log files (Windows and Ubuntu event Logs) along with bi-directional flow-based data in *.csv* file generated using *CICFlowMeter-V3* – [15] a network traffic flow generator and analyser. The CSE-CIC-IDS2018 dataset is downloaded from Amazon Web Services (AWS) following the procedure mentioned in [13].

Table 1. Overview of cse-cic-ids2018 dataset.

Name of Dataset	**CSE-CIC-IDS2018 dataset**
Year of Release	2018
Nature of Data	Packet and Bidirectional Flow based, Multiclass
Dataset Volume & Duration	~16 Million, 10 Days
No. of Features	80
No. of Classes	15
Recording Environment	Emulated, Diverse Network

We have used flow-based data stored in 10 csv files each of which contains the data collected on a single day. For each flow, 80 numbers of network traffic features are extracted including metadata such as Destination port, Timestamp and Aattack classes. The dataset includes wide range of attacks such as Brute-force, Botnet, DoS, DDoS, Web attacks, Infiltration. There are altogether 15 numbers of labels (including Benign) in the dataset which indicate different attack types along with the name of tools which are used to generate such attacks.

PRE-PROCESSING OF DATASET

CSE-CIC-IDS2018 dataset is observed to have some shortcomings which are handled in the pre-processing stage of the dataset to make it more suitable for evaluating machine learning models. The shortcomings are: [i] Traffic data spreading over 10 separate files [ii] Huge volume of data with high class imbalance [iii] Missing and Inconsistent values [iv] Duplicate records.

CSE-CIC-IDS2018 dataset contains attack information scattered in 10 separate files each of which consists of records pertaining to some specific attacks with benign traffic. As it is difficult to process each individual file for multiclass classification, all the files are merged together to form a single dataset to be used for training and testing of IDS models.

By merging all 10 files, we have the whole shape of dataset consisting of 1,62,32,943 instances and 80 features with 15 class labels (1 Benign + 14 Attack Labels). It is found that separate labels have been used in the dataset for various types of Brute Force, DoS, DDoS and Web attacks depending upon the various tools and modes used to generate those attacks. This has increased the number of labels and does not become helpful unless anyone aims to precisely differentiate between various patterns of a particular type of attack. In order to have less number of attack classes and thus reducing the class imbalance to some extent, attack types are normalised by grouping similar types of attacks into a new class and relabelled them accordingly. This has reduced the number of classes to 7 (1 Benign + 6 Attack Labels) in the merged file.

Table 2. Dataset Information After Complete Pre-Processing.

Original Label	Modified Label	No. of Instances	% of share w.r.t total instances
Benign	Benign	49,50,651	64.31%
FTP-BruteForce SSH-BruteForce	Brute Force	3,80,920	4.95%
DoS attacks-GoldenEye DoS attacks-Slowloris DoS attacks-Hulk DoS attacks-SlowHTTPTest	DoS	6,54,281	8.50%
DDoS attacks-LOIC-HTTP DDoS attacks-HOIC DDoS attacks-LOIC-UDP	DDoS	12,63,878	16.42%
Brute Force- Web Brute Force- XSS SQL Injection	Web	928	~0%
Infiltration	Infiltration	1,60,635	2.10%
Bot	Bot	2,86,176	3.71%
Total instances after pre-processing of dataset		76,97,469	100%

However, even after grouping and relabeling, it is found that the dataset is still highly imbalanced with *benign* traffic occupies large share (> 83%) of records. In reality, normal traffic will definitely outnumber the attack traffic in a big margin; however, in case of a training dataset each probable class should have comparable representation with others so that classifier can learn those uniformly without any bias towards class having majority shares which, otherwise, will lead to lower accuracy with higher false alarm. In order to improve the balance in the dataset we have removed all the duplicate records related to '*benign*' traffic while keeping those in the attack traffic.

Moreover, we have also removed all the benign records in the file "*Tuesday-20-02-2018_TrafficForML_CICFlowMeter.csv*" as it singlehandedly contributes maximum samples to the benign class and less samples for other classes. Removing them helps to bring down the class imbalance further. Due to the availability of sufficient amount of samples, we dropped samples with Infinity, NaN, missing or erroneous values from all categories. We have also removed two parameters '*Dst Port*' and '*Timestamp*' from the original feature set as they do not carry any specific information regarding the network traffic.

After pre-processing the original dataset through clean-up, merging, class reduction as discussed, we have finally a dataset with 76,97,926 instances and 78 features which includes 7 class labels (1 Benign + 6 Attack Labels). This derived dataset (Table 2) has been used for experiment with different models for multi-class detection.

CLASSIFICATION TECHNIQUES

Classification is a computational technique which identifies the particular category (among a few) of a new instance of data on the basis of knowledge developed in course of training with dataset of similar instances whose categories are known. Variety of classification methods are available in classical machine learning domain such as Linear Regression (LR), K-Nearest Neighbors (KNN), Naïve Bays, Support Vector Machine (SVM), Decision Tree, Boosting (Adaboost, XGBoost) & Bagging (Random Forest) etc. which use various mathematical techniques to make its prediction on the class of input data. Neural Network based approach is another useful classification technique which is motivated by the biological nervous system and builds up approximate functions relating input data to its probable class.

We have conducted our experiments to evaluate the multiclass classification performance of Random Forest, XGBoost and Multi Layer Perceptron (MLP) classifiers on CSE-CIC-IDS2018 dataset pre-processed as mentioned above.

Ensemble Learning

Ensemble learning [16] is a kind of supervised machine learning method where a group of predictors (often termed as "weak learners") are trained to solve the same problem and their individual predictions are suitably combined to generate better predictions. The group of predictors is called "ensembles".

Based on the process of aggregation of ensembles i.e weak learners, ensemble learning can be broadly classified in two categories namely [i] Bagging [ii] Boosting.

1. In *Bagging* or *Bootstrap Aggregation* [17] process, multiple numbers of uniform weak models are trained independently from each other in parallel with the subsets of original dataset by randomly fetching samples with replacement and finally their individual predictions are aggregated to produce final result following some kind of statistical methods such as voting, averaging etc. We have done our experiment with Random Forest (RF) [18] - a bagging technique which implements a large number of "decision trees*"* based on different samples and different feature combinations. The final prediction depends on the majority of predictions from the trees constituting the ensemble.
2. In *Boosting* process [17], uniform weak models are trained sequentially in a very adaptive way (a base model depends on the previous ones) by iteratively adjusting weights of observation from previous classification and combines them following a deterministic strategy to produce more accurate detection. There are different variants of boosting methods such as *AdaBoost (**Ada**ptive **Boost**ing), Gradient Boosting, XGBoost (e**X**treme **G**radient **Boost**ing)* etc.

AdaBoost sequentially trains new classifier to correct its predecessor by assigning a lager weight to the misclassified instances in each round of prediction. Thus the alogorithm always focuses more on the hard-to-classify samples and produces the final prediction by summing up the weighted prediction of each classifier. In *Gradient Boosting*, the procedure begins by training an initial decision tree with the given data. Then a second tree is built that aims to accurately predicting the samples where the first model performs badly. The combination of these two models is expected to have better prediction accuracy than either of the models alone. Then this process of boosting is repeated many times by addition of decision trees to the model. Each successive model attempts to minimize the prediction error of the combined boosted ensemble of all previous models.

XGBooost [19] is an advanced implementation of the gradient boosting algorithm. It is highly scalable owing to the several important system and algorithmic optimizations such as novel tree learning algorithm for handling sparse data. XGBoost, although basically a sequential boosting method, supports parallelism in lower level i.e tree construction process at each step which makes it learn faster resulting quick model creation.

As AdaBoost is a sequential learner, training this classifier on large dataset having high dimensionality is computationally much slower in comparison to parallel learners such as RF and XGBoost classifier. Hence, we have chosen RF and XGBoost and avoided AdaBoost in our experiment.

Multi Layer Perceptron

Multi Layer Perceptron (MLP) is a type of feed forward multilayer artificial neural network which is trained by iteratively adjusting weights and bias at each layer through optimizing a *loss function* (the error between the actual output and the predicted output) using gradient-based optimization and using a supervised learning technique, called *backpropagation* [20]. MLP is composed of three types of layers: [a] an input layer which receives and passes the input data to the next layer, [b] an output layer which makes final prediction or decision on the input data and [c] one or multiple hidden layers, stacked between input and output layer, which represent the important features of the problem domain and establish a non-linear mapping between input and its corresponding output. With the introduction of hidden layers in MLP, the neural network extends in depth to have more number of internal parameters which make it capable to figure out more complex non linear function describing the input and output relation. Depending on the nature of the problem, an MLP model is designed by deciding over number of hidden layers, connections between successive layers, optimizer, activation function and loss function. While training a MLP, whole training dataset is divided into number of batches - each of them contains a fixed number of training samples which is called '*batch size'*. The internal parameters of the model i.e *weights* and *biases* are updated each time the model is completely trained through a batch. In every iteration, known as '*epoch*', model encounters with all the training samples in multiple batches and keeps on adjusting the values of weights and biases. Moreover, the model is repeatedly fed with the same training dataset through multiple numbers of epochs in order to finally produce correct prediction.

PERFORMANCE METRICS

One of the key aspects of considering a machine learning model fit for its purpose is measuring its predictive ability. Various performance metrics [21] are available to evaluate the predictive ability of a model. However, choice of correct performance metrics is important to ascertain the suitability of a model in production environment.

Confusion Matrix

The confusion matrix is an n x n square matrix (n is the number of classes) where the rows represents the instances of actual classes and the columns represents the instances of predicted classes. Conventionally same order is maintained in listing the classes along the rows and columns of the matrix and therefore, elements placed along the main diagonal from top left to bottom right denotes the correctly classified instances of the dataset. Rests of elements of the matrix are the classification errors (either Type-1 or Type-2). In case of multiclass classification problems, an ideal model is supposed to produce a confusion matrix whose diagonal elements only have the non-zero values and all non-diagonals elements have zero value. This means there is no false positive or false negative and all instances are correctly classified. However, it does not normally happen in practice due to inaccuracy involved in the model. Hence, main objective remains to lower the value of both false positive and false negative as much as possible to achieve higher precision and recall [21] for each class.

Receiver Operating Characteristics (ROC) Curve

ROC is plotted by aggregating TPR (True Positive Rate or Recall) along y-axis and FPR (False Positive Rate) [21] along x-axis for all possible classification thresholds. ROC curve along with the measure of 'Area under the ROC curve (AUC)' is used as a performance comparison metric for machine learning models. If AUC is higher ($0 \leq$ AUC ≤ 1), the machine learning model is better.

Precision-Recall (PR) Curve

Precision-Recall curve is a metric which plots precision as a function of recall value across various thresholds and is used to evaluate a classifier's quality, particularly in case of hugely imbalanced dataset. The precision-recall curve summarizes confusion matrix by trading off between precision and recall. In case of highly imbalanced dataset, if the number of 'true negatives' (TN) are very high, then x-axis in ROC curve i.e FPR (false positive rate) which has TN in denominator, will have a very small value resulting a shift of ROC plot to the left side and rise up AUC score closing to 1, which is deceptive. As PR curve is constituted with precision and recall values, which do not include 'true negatives', it is more suitable than ROC for such analysis where existence of true negative does not play a major role or occurrence of true negatives are quite large due to class imbalance in dataset. A large area under the PR curve represents both high recall and precision and indicates higher accuracy in predictions of a model. As CICIDS2018 is highly imbalanced dataset, where samples of 'benign' class are significantly large in number than other attack classes, PR curve has been considered to be more useful performance metric than ROC.

A more complex Precision-Recall curve can be plotted by displaying each curve individually, along with F1-score [21] iso- curves showing the relationship between precision and recall for various F1 scores.

EXPERIMENTAL RESULTS & ANALYSIS

Experimental Platform

In this experiment, we have used "*Scikit-learn*" which is a well known free machine learning library for the Python programming language. It provides a range of supervised and unsupervised learning algorithms and is designed to interoperate with the Python numerical and scientific libraries *NumPy* and *SciPy*. We have used *Google Colaboratory* platform (*Colab* in short) for developing and executing the machine learning code in *ipython*. Google Colab is an online Jupyter notebook service that requires no configuration to use, while providing free access to computing resources including GPUs/TPUs (Tensor Processing Units) with sufficient memory and disk space. It also provides easy interface with google drive for accessing dataset and saving models.

We have conducted our experiment with two types of ensemble learning methods (RF & XGBoost) along with Multi Layer Perceptron and our findings are reported in the following sub-sections. Some common steps, as mentioned below, are followed in the experiments with the above methods to achieve the best possible results.

i. Dataset has been divided in three parts – 70% for training, 15% for validation and 15% for testing.
ii. Important hyper parameters associated with various methods are tuned to improve the performance of the model till the performance metrics shows no more improvement specially in cases of minority classes.
iii. Three types of performance metrics namely Confusion Matrix, ROC-AUC curve and Precision-Recall (PR) curve (with iso-f1 lines) are presented to evaluate the models.

Experimental Observations & Analysis

Random Forest (RF)

Table 3 displays the experimental results with RF method in which 'Decision Tree' is the base classifier. Hyper parameters like number of estimators' i.e number of decision trees (*n_estimators*), maximum size of each tree (*max_depth*) and class weight (*class_weight*) are tuned to achieve better performance especially in respect of the minority classes i.e Infiltration & Web. To find the best number and size of decision trees in the RF ensemble for optimal result, test dataset has been fed to the classifier with various combinations of *n_estimators* values such as 30, 60, 80, 100, 120 and *max_depth* values such as 4, 6, 8, 12, 16, 20. We have used three types of values for *class_weight* parameters in RF classifier function. By default "*None*" value is used which associates same weight '1' to all classes. "Balanced" mode is used to assign each class with varying weights inversely proportional to class frequencies in the input data following the relation $\mathbf{w_i = (n/kn_i)}$ (where n = total number of instances, k = number of classes, $n_i =$ class frequency). We have also conducted experiments with customised class weights for each class given in the same sequence of class labels by providing each sample of the training dataset with a custom weight obtained by tuning the class weight used in balanced mode and hence not in strict inverse proportion to the class frequencies.

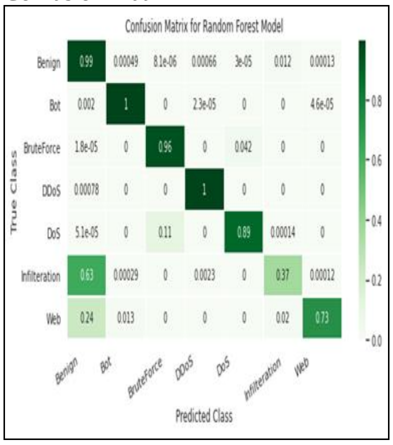
It is observed that high recall i.e TPR (89% -100%) along with high precision (83%-100%) has been achieved for Bot, Bruteforce, DDoS & DoS types of attacks using *n_estimators* as 80 and *max_depth* as 20 irrespective of *class_weight* chosen, which means values of flow parameters in the traffic generated by the each of those attacks are fairly distinguishable not only from 'Benign' traffic, but also from each other. Areas under PR curves of them are also close to 1. 'Infiltration' has very low TPR with low precision owing to many of its records being misclassified mostly as "Benign". This is due to very little difference between its traffic patterns with the normal traffic, which is beyond the detection ability of the RF detector model used. 'Web' class is found to have low precision value with higher recall (than 'Infiltration') which makes its F1-score moderately high (>70%). It is also observed that the precision & recall of 'Infiltration' and 'Web' classes are largely affected by varying the class weights.

It is evident from the plotting of multiclass PR curves for customised class-weights that the RF classifier is equally good in detecting attacks like Bot, Bruteforce, DDos & DoS as their areas under the curve being very close to 1, whereas precision of the model sharply declines with the increase of recall in case of Infiltration and Web attack. For Infiltration, precision falls more sharply.

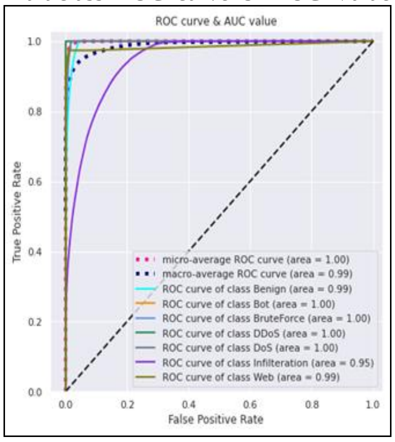
Table 3. Random Forest

Hyper Parameters: n_estimators=80, max_depth=20, class_weight = {Customised weights for classes}

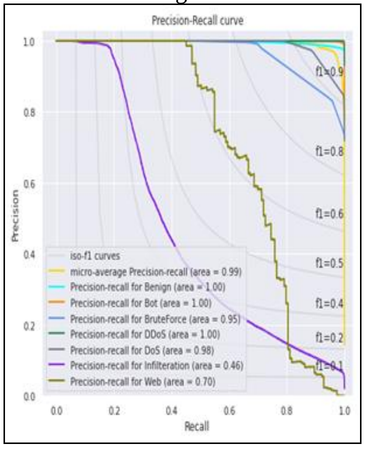
Confusion Matrix



Multiclass ROC curve & AUC Value



PR Curve & Avg. Precision Score



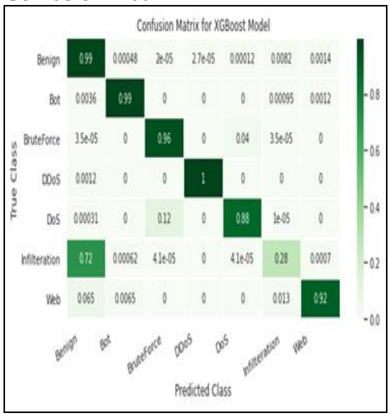
recision and Recall Score: Overall Test Accuracy: 97%

	Benign	Bot	BruteForce	DDoS	DoS	Infiltration	Web
Precision	0.98	0.99	0.83	1.00	0.97	0.50	0.52
Recall	0.99	1.00	0.96	1.00	0.89	0.37	0.73

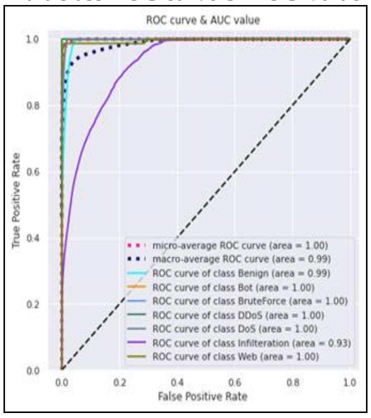
Table 4. XGBOOST.

Hyper Parameters : n_estimators= 20, max_depth=8, class_weight= {Customised weights for classes}

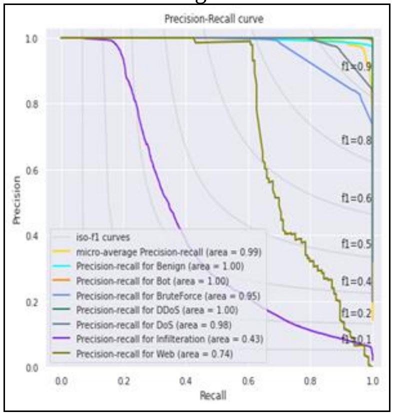
Confusion Matrix



Multiclass ROC curve & AUC Value



PR Curve & Avg. Precision Score



Precision and Recall Score: Overall Test Accuracy: 97%

	Benign	Bot	BruteForce	DDoS	DoS	Infiltration	Web
Precision	0.98	0.99	0.83	1.00	0.97	0.52	0.12
Recall	0.99	0.99	0.96	1.00	0.88	0.28	0.92

XGBoost

In Experiment with XGBoost, hyper parameters *n_estimators, max_depth and class_weight* are tuned similarly as in RF model. However, in comparison with RF, XGBoost produced much better results with less number of decision trees i.e estimators (20) and maximum size of each tree (*max_depth*) (8) as depicted in the Table 4. Tuning of *class_weight* parameter played an important role where much better result has been obtained by using different customized weights to each class rather than using balanced (uniform) class weights for all classes. We have used *'Dart'* as booster parameter in this model to drop out trees to avoid over-fitting issue [22]. It is observed that apart from giving high TPR (88%-100%) and high precision (83%-99%) in detecting Bot, Bruteforce, DDos & DoS types of attacks with less number of estimators and smaller tree size, XGBoost classifier shows its higher efficacy in classifying 'Web' based attack (a severely minority class) with 92% TPR value. However, XGBoost fails to efficiently distinguish 'Infiltration' type of attack with TPR value being only 28%.

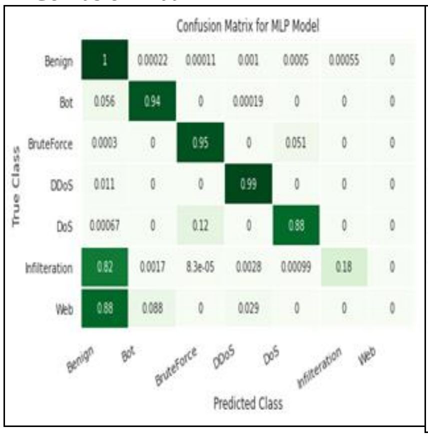
Multi Layer Perceptron (MLP)

In the experiment with deep neural network, a MLP model has been optimally designed with one input layer, two hidden layers and one output layer. All the input and hidden layers are having 78 numbers of neurons (equal to the number of features) whereas the output layer consists of 7 numbers of neurons (i.e equal to the number of classes). *'LeakyRelu'* and *'Softmax'* are respectively used as *activation functions* in the hidden layers and output layer. The *'categorical cross entropy'* and *'adam'* methods are chosen as *loss function* and *optimizer* respectively in this model. The said model has been finalized after several trials with various methods of activation functions, loss functions and increasing number of hidden layers along with variable number of neurons in different layers and drop-out values. The final model has been trained with different values of two hyper parameters - *epoch* and *batch size*. The best result is observed with *epoch value = 80* and *batch size = 65,536*. Multiclass classification capability of the MLP model is shown in the Table 5. It is evident from the table that the ROC-AUC curve does not corresponds with confusion matrix in respect of minority classes like Infiltration and Web. Because, confusion matrix shows that true positive rate or recall values of those two classes are very low (18% for Infiltration and 0% for Web) where as their AUC values in ROC curve is quite high (0.9 for Infiltration and 0.97 for Web). On the other hand, AUC scores of each class in PR curve with iso-F1 lines are consistent with the confusion matrix. Hence, the evaluation result expressed by ROC curve in this case is also found to be misleading particularly in case of poorly represented classes and PR curve is a evidently a better choice in this case.

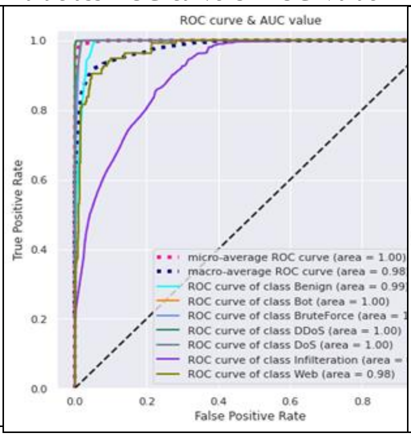
Table 5. Multi Layer Perceptron (MLP).

Hyper Parameters: epochs=80,batch_size=65536

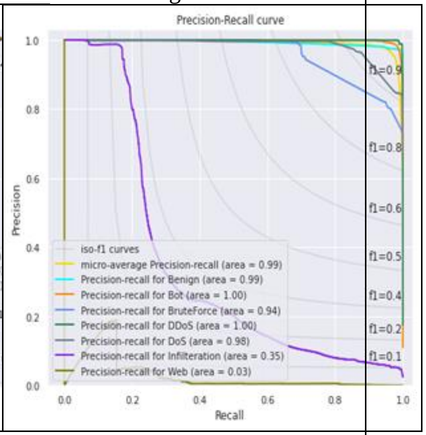
Confusion Matrix



Multiclass ROC curve & AUC Value



PR Curve & Avg. Precision Score



Precision and Recall Score: Overall Test Accuracy: 96%

	Benign	Bot	BruteForce	DDoS	DoS	Infiltration	Web
Precision	0.97	0.99	0.82	1.00	0.96	0.91	0
Recall	1.00	0.94	0.95	0.99	0.88	0.18	0

CONCLUSION & FUTURE OBJECTIVES

The reported experiment with Ensemble and MLP methods using a large multi-class imbalanced dataset like CSE-CIC-IDS2018 validates some of the key points such as [i] Assigning non uniform customised class-weights to the various classes depending upon their share in a imbalanced dataset produces better results specially for the poorly represented class [ii] For a structured and highly imbalanced dataset, ensemble classifier out performs MLP technique in detecting minority classes [iii] XGBoost classifier is comparatively much faster and more efficient than RF as it gives out better performance with less number of decision trees and less amount of depth of trees. [iv] PR plot with iso-F1 curve is more authentic performance metrics than ROC plot with AUC score in case of a imbalanced dataset where a particular class has disproportionally higher share in the training dataset which results in high true negative value forcing FPR to a very low value making high AUC score which is deceptive.

CSE-CIC-IDS 2018 is an emulated dataset created in a test environment where malicious traffic has been synthesised using some known attack tools. Such dataset is not supposed to have all the nuances of malware traffic in real scenario for obvious reasons. Hence, we aim to develop a ML model trained with a dataset generated from real network traffic captured within a production network environment.

Acknowledgement
The authors sincerely acknowledge the contribution of Shri Sambudhha Roy, an engineering student of Heritage Institute of Technology, Kolkata, at initial stage of this work. Authors are also grateful to Dr. Sumit Som, Director, VECC for his support and encouragement for this activity.

REFERENCES

1. IDS 2018 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. 2018. Available from: https://www.unb.ca/cic/datasets/ids-2018.html
2. Ring, Markus & Wunderlich, Sarah & Scheuring, Deniz & Landes, Dieter & Hotho, Andreas. (2019). “A Survey of Network-based Intrusion Detection Data Sets.” Computers & Security. 86. 10.1016/j.cose.2019.06.005.
3. Fitni, Q. R. S., & Ramli, K. (2020). “Implementation of ensemble learning and feature selection for performance improvements in anomaly-based intrusion detection systems.” *In Proceedings - 2020 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology, IAICT 2020 (pp. 118-124).*
4. G. Karatas, O. Demir and O. K. Sahingoz, "Increasing the Performance of Machine Learning-Based IDSs on an Imbalanced and Up-to-Date Dataset," in *IEEE Access*, vol. 8, pp. 32150-32162, 2020, doi: 10.1109/ACCESS.2020.2973219.
5. Chawla, Nitesh & Bowyer, Kevin & Hall, Lawrence & Kegelmeyer, W.. (2002). “SMOTE: Synthetic Minority Over-sampling Technique.” J. Artif. Intell. Res. (JAIR). 16. 321-357. 10.1613/jair.953.
6. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. (2017). “LightGBM: A Highly Efficient Gradient Boosting Decision Tree.” NIPS.
7. Y. Hua, "An Efficient Traffic Classification Scheme Using Embedded Feature Selection and LightGBM," *2020 Information Communication Technologies Conference (ICTC)*, Nanjing, China, 2020, pp. 125-130, doi: 10.1109/ICTC49638.2020.9123302.
8. Rumelhart, D., Hinton, G. & Williams, R. “Learning representations by back-propagating errors.” *Nature* **323,** 533–536 (1986).
9. Basnet, Ram & Shash, Riad & Johnson, Clayton & Walgren, Lucas & Doleck, Tenzin. (2019). “Towards Detecting and Classifying Network Intrusion Traffic Using Deep Learning Frameworks.” 10.22667/JISIS.2019.11.30.001.
10. Filho, Francisco & Silveira, Frederico & Junior, Agostinho & Vargas-Solar, Genoveva & Silveira, Luiz. (2019). “Smart Detection: An Online Approach for DoS/DDoS Attack Detection Using Machine Learning.” Security and Communication Networks. 2019. 1-15. 10.1155/2019/1574749.
11. LeCun Y., Haffner P., Bottou L., Bengio Y. (1999) “Object Recognition with Gradient-Based Learning. In: Shape, Contour and Grouping in Computer Vision.” Lecture Notes in Computer Science, vol 1681. Springer, Berlin, Heidelberg

12. Kim, Jiyeon & Kim, Jiwon & Kim, Hyunjung & Shim, Minsun & Choi, Eunjung. (2020). “CNN-Based Network Intrusion Detection against Denial-of-Service Attacks.” Electronics. 9. 916. 10.3390/electronics9060916.
13. Kanimozhi, V. & Jacob, Prem. (2019). “Artificial Intelligence based Network Intrusion Detection with Hyper-Parameter Optimization Tuning on the Realistic Cyber Dataset CSE-CIC-IDS2018 using Cloud Computing.” 0033-0036. 10.1109/ICCSP.2019.8698029.
14. R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305-316, doi: 10.1109/SP.2010.25.
15. Applications | Research | Canadian Institute for Cybersecurity | UNB. 2017. Available from: https://www.unb.ca/cic/research/applications.html
16. Dietterich T.G. (2000) “Ensemble Methods in Machine Learning. In: Multiple Classifier Systems.” MCS 2000. Lecture Notes in Computer Science, vol 1857. Springer, Berlin, Heidelberg.
17. Bühlmann, Peter. (2012). Bagging, Boosting and Ensemble Methods. Handbook of Computational Statistics. 10.1007/978-3-642-21551-3_33.
18. Tin Kam Ho, "Random decision forests," Proceedings of 3rd International Conference on Document Analysis and Recognition, 1995, pp. 278-282 vol.1, doi: 10.1109/ICDAR.1995.598994.
19. Chen, T.Q. and Guestrin, C. (2016) XGBoost: A Scalable Tree Boosting System. arXiv:1603.02754v3.
20. LeCun, Yann & Bengio, Y. & Hinton, Geoffrey. (2015). Deep Learning. Nature. 521. 436-44. 10.1038/nature14539.
21. Grandini M, Bagli E, Visani G. Metrics for multi-class classification: an overview. arXiv preprint arXiv:2008.05756. 2020 Aug 13.
22. Rashmi, K. and Ran Gilad-Bachrach. “DART: Dropouts meet Multiple Additive Regression Trees.” ArXiv abs/1505.01866 (2015): n. pag.