Tabriz Journal of Electrical Engineering (TJEE), vol. 52, no. 3, Autumn 2022 Serial no. 101
DOI: 10.22034/tjee.2022.15567

An MLP-based Deep Learning Approach for Detecting DDoS Attacks

M. Vasou Jouybari[1], E. Ataie[2*], M. Bastam[2]

[1] Department of Computer Science, University of Sistan and Baluchestan, Zahedan, Iran. E-mail: mojtaba.vasou@pgs.usb.ac.ir
[2] Department of Computer Engineering, University of Mazandaran, Babolsar, Iran. . E-mail: ataie@umz.ac.ir
[3] Department of Computer Engineering, University of Mazandaran, Babolsar, Iran. . E-mail: bastam@umz.ac.ir

[*]Corresponding author

Received: 05/15/2022, Revised: 09/14/2022, Accepted: 10/26/2022.

Abstract
Distributed Denial of Service (DDoS) attacks are among the primary concerns in internet security today. Machine learning can be exploited to detect such attacks. In this paper, a multi-layer perceptron model is proposed and implemented using deep machine learning to distinguish between malicious and normal traffic based on their behavioral patterns. The proposed model is trained and tested using the CICDDoS2019 dataset. To remove irrelevant and redundant data from the dataset and increase learning accuracy, feature selection is used to select and extract the most effective features that allow us to detect these attacks. Moreover, we use the grid search algorithm to acquire optimum values of the model's hyperparameters among the parameters' space. In addition, the sensitivity of accuracy of the model to variations of an input parameter is analyzed. Finally, the effectiveness of the presented model is validated in comparison with some state-of-the-art works.

Keywords
Distributed denial of service, Network security, Machine learning, Multi-layer perceptron, CICDDoS2019.

1. Introduction
Denial of Service (DoS) attacks, whose goal is to prevent legitimate users from gaining access to a particular network resource, have been recognized by the network research community since the early 1980s. The first Distributed Denial of Service (DDoS) attack case was noticed by the Computer Incident Advisory Capability (CIAC) in the summer of 1990. Usually, hackers use botnets (the networks formed by enslaving devices) worldwide to manage a DDoS attack. Internet of Things (IoT) devices are rapidly expanding such that they reached from two billion devices in 2006 to more than eight billion devices in 2020 [1, 2]. Because of the lack of network security in these devices, they are stealthily used as massive IoT botnets. An attacker may use several malware injection tools to take control of such devices [3]. Computers and IoT devices that form a botnet, can launch massive DDoS attacks. DDoS attacks are classified into several categories, such as amplification attacks (DNS, NTP, SNMP, etc.), floods (UDP, ICMP, SYN, etc.), IP fragmentation, and zero-day attacks [4]. DDoS attackers usually concentrate on bandwidth, network protocols, and network and application layers which are typically measured in terms of bits per second, packets per second, and requests per second, respectively [5, 6].

Several detection methods come from various theories and models, such as information theory, statistical models, and machine learning. The methods mentioned above, are the three main approaches proposed in DDoS attack detection investigation associations, such as USENIX [7] and ISACA [8]. These methods form the foundation of the most recent detection strategies [9–11]. Information theory models usually suffer from limitations of classical sets, and negligence of semantics and practical use of information [12]. Moreover, existing detection systems based on statistical anomalies are limited because they need to assign thresholds to detect. Using machine learning techniques, network intrusion detection systems are able to overcome limitations of the solutions proposed based on other methods [13, 14].

Machine learning has been known as a helpful cybersecurity strategy by introducing the right plan for analysing and performing the right action automatically [15]. It includes several techniques, such as Artificial Neural Networks (ANNs), Support Vector Machines (SVMs), Logistic Regression (LR), Bayesian networks, Decision Trees (DTs), clustering, ensemble learning, etc [16]. Different ANN models have been utilized in the field of Intrusion Detection Systems (IDSs). ANNs can play a critical role in detecting DDoS attacks due to several intrinsic characteristics, including self-organizing, self-learning, robustness, parallelism, and fault tolerance [17]. Deep learning models are designed with structures like ANNs that can learn and make intelligent decisions. They show reasonable results in distinguishing attack traffic from legitimate traffic based on specific features [13, 14]. In this paper, the objective is to detect anomalies in the input traffic of a network through the proposed model

DOI: 10.22034/tjee.2022.15567

based on Multi-layer Perceptron (MLP). MLP models are ANN models that can detect zero-day DDoS attacks if they are trained with a similar protocol to the unknown attacks [18]. As the network traffic and DDoS attacks become more complex, some features of the original dataset may become unable to individualize network traffic. So, detector errors could increase when traffic changes [16]. In order to enhance the accuracy of the proposed model and reduce the computation requirements, feature selection is used to remove redundant and irrelevant features to deal with this problem. Moreover, the grid search algorithm is exploited to increase accuracy of the presented MLP model by running it with a combination of all hyperparameters separately and obtaining optimum parameters. The presented model is trained and tested by the CICDDoS2019 dataset [4]. The result of experimental analysis shows that the presented model can not only detect advanced DDoS attacks better than the state-of-the-art algorithms, but it can also perform with high accuracy and reduce false negative and false positive rates. The impact of variation of an input hyperparameter of the MLP model on the accuracy of the proposed model is also investigated in this study.

The rest of this paper is organized as follows. In Section 2, we discuss related work. The proposed model is presented in Section 3. Performance evaluation metrics are introduced in Section 4. We elaborate our experiments on the CICDDoS2019 dataset in Section 5. In Section **Error! Reference source not found.**, the results of the proposed model are compared with a few state-of-the-art algorithms including deep and shallow machine learning methods. Sensitivity analysis is investigated in Section 7. Finally, in Section 8, conclusions are presented and recommendations regarding future research work are given.

2. Related work

Various methodologies and techniques have been proposed to detect DDoS attacks. This section reviews the research efforts made for detecting DDoS attacks primarily based on six distinct ANN models, including MLP, Stacked Autoencoders (SAE), entropy approaches (deep learning and genetic algorithm), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Random Forest (RF).

Saied et al. [19] have developed a mechanism to detect DDoS attacks using an ANN algorithm based on modelling particular features that can separate attack traffic from legitimate network traffic. The authors proposed three types of MLP algorithms with different nodes. Each MLP algorithm with a specific structure can identify one type of DDoS attack, including TCP, UDP, and ICMP attacks. The main idea of their study is to investigate both known and unknown DDoS attacks, then create a defence system to stop malicious packets before reaching the victim machines and let the original packets pass without dropping. The authors launched 580 known and 580 unknown attacks, and the proposed method was able to detect 95% of unknown and 100% of known DDoS attacks.

Wang et al. [16] have used a dynamic MLP classifier as a solution to identify DDoS attacks. This model consists of three modules, including knowledge base, detection model, and feedback mechanism. The authors exploited a wrapper feature selection called sequential backward selection in the training phase to select optimal features. Also, they designed a feedback mechanism to reconstruct the detector after receiving the detection errors dynamically.

Sumathi et al. [20] have proposed a deep learning neural network to detect DDoS attacks. In this method, network performance is evaluated using a deep neural network classifier with the strategy of minimizing cost for a publicly available dataset. The authors used the KDD dataset, and a mixed dataset consisting of Conficker, CAIDA, and UNINA datasets. The efficiency of the presented model was assessed in terms of average delay, overhead, detection accuracy, packet delivery ratio, packet loss, cost per sample, and throughput. The primary purpose of the study was to reduce classification error rate and make DDoS detection more accurate.

Niyaz et al. [21] have detected multi-vector DDoS attacks using SAE based on the deep learning model. The authors run the proposed model at the network layer in the SDN. They used a set of features extracted from network packet headers and then decreased this set of features with an unsupervised model through a deep learning technique. They implemented the presented model on a set of traffic data collected from various circumstances. The proposed approach attempted to detect DDoS attacks on both the control plane and the data plane of the SDN and was also fully implemented in the SDN controller.

Ujjan et al. [22] have developed an adaptive polling-based sampling and sFlow with Snort IDS and also a deep learning-based model, which reduces the variety of common DDoS attacks in IoT networks. The team was able to program the parameters needed for network devices that did not need to use third-party software or hardware because of the flexibility of separation in SDN. In the first phase of this study, the authors developed polling-based sampling and sFlow separately at the data plane to decrease the processing and network overhead of the switches. In the second phase, they developed Snort IDS in collaboration with the SAE deep learning model to optimize detection accuracy at the control plane.

Singh et al. [23] have presented an approach to detect DDoS attacks at the application layer using a multi-layer perceptron classification algorithm. They also applied correct weights of connection of layers of MLP model using a genetic algorithm. The proposed model was exploited to identify DDoS attacks based on the number of HTTP-GET requests, entropy of requests, and entropy variance per IP address. However, it was shown that the amount of entropy could be higher regarding normal client cases and less in attack cases.

He et al. [24] have studied network-based DDoS attacks initiated from the cloud. The authors designed a new system that detects and mitigates DDoS attacks on the source side of the cloud. To do this, they used nine machine learning algorithms categorized into supervised and unsupervised algorithms. The supervised ones include random forest, linear regression, decision tree, SVM (with linear, radial basis function neural network, or polynomial kernels), and Naive Bayes algorithms. In contrast, unsupervised ones include k-means and the Gaussian model. Then, a prototype was implemented and

DOI: 10.22034/tjee.2022.15567

tested in a cloud environment to compare the results obtained from the aforementioned algorithms.

Doriguzzi-Corin et al. [14] have developed a lightweight and practical deep learning system called LUCID to detect DDoS attacks. The system classifies network traffic flows as attack or benign ones according to the characteristics of the CNNs presented in the paper. The authors used a novel method to pre-process input data to support the online attack detection system. Since the primary purpose of the study was to minimize the complexity and execution time of the resources used in the presented CNN model, a lightweight and supervised detection system that includes CNN was exploited. Such a CNN model, unlike statistical detection methods, did not need to adjust thresholds, and it could reduce feature engineering efforts and the need to human experts. Using the most recent datasets, LUCID increases processing speed 40 times compared to baseline methods.

Yuan et al. [13] have developed a detection method for DDoS attacks using deep learning, including CNN, RNN, and fully connected layers. The deep learning method was chosen because of its ability to automatically extract high-level features from lower-level features. Given the capability of RNN to learn from historical network packets, the authors used RNN instead of using conventional machine learning methods. Then, they proved that RNN performed better than random forest in general and tracked network attack activities and used network traffic sequences to learn patterns. They reduced the error rate from 7.517% to 2.103% in comparison with random forest as a conventional machine learning approach in a large dataset.

Sanchez et al. [25] have utilized traditional machine learning methods in developing a DDoS detector software. To optimize the detection capability of each machine learning model, the authors applied an exhaustive hyperparameter search based on the hyperparameters of each algorithm. Using this grid search approach, the performance of different methods was evaluated when different datasets were used for training and testing the models. The results showed that tuning hyperparameters in traditional machine learning allows for increased performance similar to deep learning approaches, but with less required resources.

Batchu et al. [26], have designed an automatic DDoS detection methodology based on hybrid feature selection and hyperparameter tuning. Both optimal features and hyperparameters were fed into several learning approaches, including SVM, LR, DT, Gradient boost (GB), and K-nearest neighbour (KNN). The experiments were evaluated on the CICDDoS2019 dataset. The steps of the proposed methodology were trained and tested in four different cases with various scenarios. In each case, the models were analysed using combinations of with/without balancing data, with/without feature selection, and with/without hyperparameter tuning. The results showed that GB model outperformed the others with an accuracy of 99.89% in case of imbalanced dataset.

Ismail et al. [27] have developed and validated a DDoS detection tool using Python based on UNSW-nb15 dataset. To this end, random forest and XGBoost algorithms were exploited and implemented for classification. The algorithms were compared based on several metrics, such as precision, accuracy, and recall. Mihoub et al. [28] have proposed a multi-class classifier based on looking-back concept for DDoS detection in IoT environments. After detection, another component, known as DDoS mitigation, is applied - based on specific packet type of the detected attack– to mitigate the attack. The proposed tool is successfully evaluated on Bot-IoT dataset.

Ghasabi et al. [1] have proposed a DDoS detection and mitigation approach which takes advantage of unique features of the SDN architecture. To do this, a statistical method based on Jeffrey distance is used. The proposed method was evaluated by Mininet simulation. Alidoosti et al. [29] have proposed a dynamic and black-box vulnerability analysis approach that can identify business logic vulnerabilities of a web application against flooding DoS attacks. The presented approach has been conducted on four open source web applications and was able to successfully detect DoS-related business layer vulnerabilities.

3. Proposed method

In this section, the proposed model for DDoS attack detection is presented. Fig. 1 shows a block diagram illustrating the major steps for constructing the proposed model.

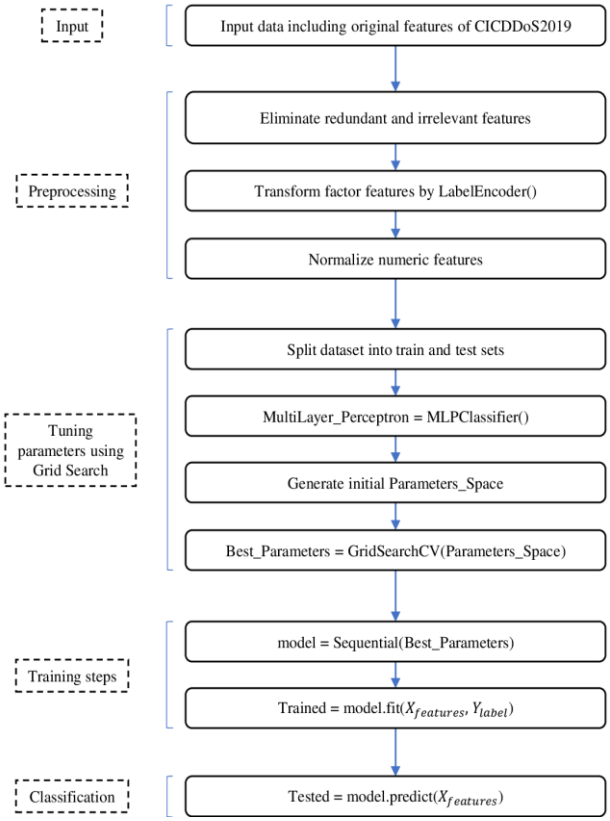

Fig. 1. The steps for constructing the proposed model.

We used a MLP, also called feed-forward neural network, as a classifier. Perceptron is the foundation of MLP architecture that is inspired by neurons of the brain. Perceptron conducts the inputs from the previous layer to the output layer after performing a mathematical operation [30]. Equation (1) shows the mathematical operation of a perceptron:

DOI: 10.22034/tjee.2022.15567

$$Y_k(x) = f\left\{\sum(w_{ki}x_i) + b_k\right\} \quad (1)$$

where, Y_k is the output of k^{th} perceptron; w_{ki} is the weight matrix of the k^{th} column and the i^{th} row of neurons; x_i is the i^{th} neuron of the input layer; b_k is the bias of the k^{th} layer of the neural network; and f is the activation function. The structure of a typical artificial neuron and the topology of the proposed MLP are shown in Fig. 2 andFig. 3, respectively [31].

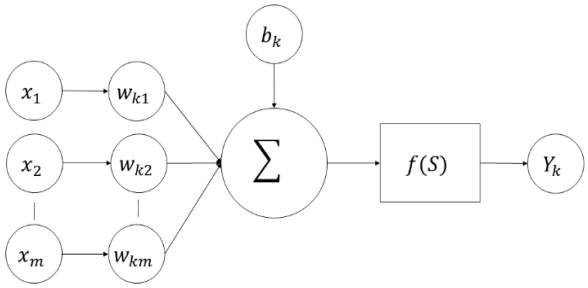

Fig. 2. A typical artificial neuron.

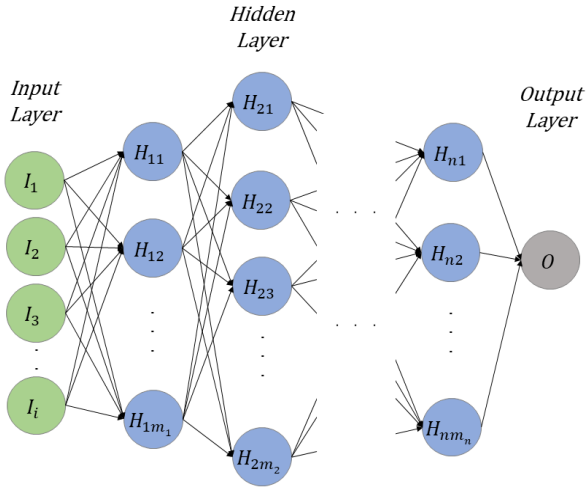

Fig. 3. Topology of the proposed MLP.

To launch an artificial neural network, the weights and biases should be adjusted. Then, an activation function for each hidden and output layer should be set up. The first step is to initiate weights and biases randomly. Depending on the number of network features, we have neurons in the input layer to train the neural network model. In this study, we used backward propagation of errors to adjust weights and biases, in order to activate hidden neurons with appropriate values. Indeed, the backward pass allows the model to modify weights and biases if an incorrect output is obtained [32]. As stated earlier, the activation function must be defined for each of the hidden and output neural network layers. The most widely used activation functions utilized in this research study, are listed in Table 1. In this study, those activation functions that provide the best performance for the proposed model, will be selected automatically by the grid search algorithm.

Table 1. The activation functions used in this research study.

Activation function	Formula
Sigmoid	$f(x) = {}^{1}/_{e^{-x}}$
Tanh	$f(x) = 2sigmoid(2x) - 1$
ReLU	$f(x) = \max(0, x)$
Softmax	$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$ $for\ j = 1,\ldots,K$

The Loss Function, also called the cost function, of the model has to be minimized by definition. In fact, to minimize the prediction error of each row of the dataset through iteratively improving the weights, the loss function is used. In the training phase and at each iteration, data is fed forward to the network and the prediction error is calculated. Moreover, the error is backward propagated through the network, and then the biases and weights are adjusted. Due to the existance of two classes of benign and attack, we used Binary Cross Entropy loss function. The loss function formula used in this reseach study over a batch of D data is as follows.

$$LF = -\frac{1}{D}\sum_{i=1}^{D} y_i.log\,\hat{y}_i + (1 - y_i).log(1 - \hat{y}_i) \quad (2)$$

where y_i is the label of the i^{th} flow in the batch of D samples, and $\hat{y}_i$ is the i^{th} predicted probability of class of benign or malicious.

The primary purpose of feature selection is to obtain a subset of the features from the main problem. Feature selection also reduces the search space defined by features. Hence, it causes an increase in the learning rate and reduces memory consumption. In the next preprocessing step, we begin to normalize these features to make it easier for the model to process the input entered into the deep neural network model faster and more precisely [33].

When the running time of the algorithm is not a priority, the recommended method of searching for the optimum parameters is the grid search. For each hyperparameter, the user provides a set of values that appears to perform well for the neural network model. Grid search trains the MLP model with each permutation of these hyperparameter values individually. Finally, it returns the best combination of hyperparameters values among the others [[35], [35]]. In this study, we have adjusted the MLP algorithm hyperparameters using grid search. In fact, the optimum solution case is obtained when the best adjustment of parameters is found after combining all parameter values. Grid search returns optimum values of each parameter to have the best performance of the MLP algorithm after combining all the parameter values separately.

4. Performance Metrics

Herein, the performance measures of interest to evaluate the proposed method are presented. Though these metrics are usually standardized, to remove any ambiguity, each metric is introduced in the following. These metrics are computed based on the confusion matrix shown in Table 2. TP refers to the sample data that are correctly identified attacks. FP refers to the sample data that are normal but misclassified as attack. TN refers to the sample data that

DOI: 10.22034/tjee.2022.15567

are normal and correctly classified as non-attack. FN refers to the sample data that are attack and wrongly identified as normal.

Table 2. The confusion matrix structure.

Predicted	Actual	
	Normal	*Attack*
Normal	True Negative (TN)	False Negative (FN)
Attack	False Positive (FP)	True Positive (TP)

Accuracy denotes the percentage of accurately classified or predicted records to the total of given records [36]. This measure is computed by Equation (3).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

Recall indicates the percentage of all accurately classified or predicted positive records to all real positive records in the dataset and is computed by Equation (4).

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

Precision is defined as the percentage of all accurately classified or predicted positive records to all predicted positive records. This measure is computed by Equation (5).

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

F1-score is a weighted mean of recall and precision, which is computed by Equation (6).

$$F1\_score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (6)$$

ROC curve stands for the receiver operating characteristic curve, which is useful in evaluating the quality of classification models used in machine learning and data mining. ROC curve is based on plotting the True-Positive Rate (TPR) on the y-axis versus the False-Positive Rate (FPR) on the x-axis [37].

$$TPR = \frac{TP}{TP + FN} \quad (7)$$

$$FPR = \frac{FP}{TN + FP} \quad (8)$$

In binary classification, each record of the test dataset has a probability value. A varying parameter, e.g., T, is defined as a threshold, so that if the probability of each record is greater than T, the record is classified as a positive sample. Otherwise it is classified as a negative one. If the instance is actually positive, $f_1(x)$ is calculated as the probability density function and if the instance is actually negative, $f_0(x)$ is calculated. Generally, the ROC curve is defined by a parametric definition as follows:

$$TPR(T) = \int_T^{\infty} f_1(x)dx \quad (9)$$

$$FPR(T) = \int_T^{\infty} f_0(x)dx \quad (10)$$

Area Under the ROC Curve (AUC) measures the entire area underneath the ROC curve. It indicates the potential of the classifier to distinguish between classes correctly [38].

5. Experimental Setup

The proposed model has been implemented in the Python programming language using TensorFlow [39], Keras [40], NumPy [41], and Scikit-Learn [42] libraries. TensorFlow is a library to train and execute very large neural network models efficiently. Keras is a high-level deep learning API and is used to make it simple to train and execute neural network models [43]. NumPy is an array programming library that is commonly used in the Python language. Scikit-Learn is a Python module for machine learning that provides efficient functions for data preparation, post-model analysis, and evaluation. For experimentations, the tests have been executed on an Intel® Xeon® CPU E5-2650 v4 with 2.20 GHz, 40 processors, and 100 GB RAM.

Table 3. The distribution of DDoS attacks inside CICDDoS2019 and statistic of training and testing sets.

Attacks				**Training set No. of packet flows**	**Testing set No. of packet flows**
DDoS Attacks	**Reflection Attacks**	**TCP based attacks**	MSSQL	4522492	5787453
			SSDP	2610611	-
		TCP/UDP based attacks	SNMP	5159870	-
			DNS	5071011	-
			NETBIOS	4093279	3657497
			LDAP	2179930	1915122
			PORTMAP	-	186960
		UDP based attacks	CharGen	-	-
			NTP	1202642	-
			TFTP	20082580	-
	Exploitation Attacks	**TCP based attacks**	SYN Flood	1582289	4891500
		UDP based attacks	UDP Flood	3134645	3867155
			UDP-Lag	366461	1873
Total Attack Flow				**50063112**	**20364525**

DOI: 10.22034/tjee.2022.15567

The most popular datasets that are released in the field of intrusion detection are DARPA [[44], [45]], KDD'99 [46], CAIDA [[47], [48]], NSL-KDD [49], ISCX-IDS-2012 [50], and CICIDS2017 [51], among them, this research is implemented using the CICDDoS2019 dataset. Table 3 summarizes the distribution of different DDoS attacks in this dataset [4]. Unlike older datasets, CICDDoS2019 includes modern attacks such as PortMap, DNS, UDP-Lag, MSSQL, and SYN. In this dataset, twelve different types of attack, including LDAP, NTP, SYN, NetBIOS, DNS, UDP, UDP-Lag, MSSQL, SSDP, SNMP, TFTP, and WebDDoS have been considered for training day, and seven types of attacks, including LDAP, SYN, UDP, NetBIOS, UDP-Lag, MSSQL, and PortScan have been considered for testing day. In CICDDoS2019, a new attack taxonomy based on the TCP/UDP protocol at the application layer was also created. The statistic of training and testing sets are shown in Table 3.

The CICDDoS2019 dataset includes raw data containing network traffic in the form of Pcap files. Moreover, it contains event logs (Ubuntu and windows event logs) for every machine. In this dataset, 87 features are extracted using CICFlowMeter-V3 software [52] and are saved as CSV files. This dataset is publicly available [53].

Two sample attack files named DrDoS_DNS and DrDoS_NTP are selected from CICDDoS2019 randomly. Afterwards, several experiments are run on these sample files. Table 4 and Table 5 show the results of these tests in terms of *accuracy*, *AUC*, and *F1-score*. In the first row of these two tables, the results of executing the proposed model in this study using the grid search (GS) on the DrDoS_DNS and DrDoS_NTP sample files are shown. In this case, the proposed model is executed without feature selection, and just the GS algorithm is run to find the best parameters of the model. It can be seen that despite the relatively acceptable results in accuracy and F1-score metrics, running the proposed model with just GS shows poor performance in the AUC metric. In fact, AUC is an important measure when the dataset is not balanced. Because the dataset under study is highly imbalanced, i.e., the number of attack samples are much more than the number of benign ones, obtaining high accuracy values is not enough. As another experiment, the proposed model is executed with both FS and GS techniques. As shown in the second row of Table 4 and Table 5, not only the accuracy and F1-score are good, but also AUC is much better than the case GS was only applied. The same results have been reported in the literature when both FS and GS techniques were used. Therefore, in the experiments performed in this study, we used both techniques.

6. Numerical Results

In this section, the performance of the proposed model on the CICDDoS2019 dataset is evaluated based on the metrics described in Section 4. The feature selection phase in the proposed method yields six features of the CICDDoS2019 dataset which seem to be most relevant to DDoS attacks. These features include: Min Packet Length, URG Flag Count, Inbound, Protocol, Fwd Packet Length Mean, and Init_Win_bytes_forward. Before feature selection, under-sampling technique is exploited to alleviate the problem of imbalanced dataset by keeping all samples of benign class (the rare class) and reducing the size of attack class (the abundant class).

The best hyperparameter setting of the proposed MLP after applying the grid search phase is as follows: batch size is 1024; hidden layer topology is (12, 32, 32, 16, 9); activation is set to "relu"; "adam" is chosen as the solver; alpha is set to 0.001; and learning rate is constant. Moreover, the maximum iteration is set to 200. Early stopping is also used to terminate training when validation score is not improved in 5 iterations.

Because the number of attack and normal classes are imbalanced in the CICDDoS2019 dataset, most DDoS detection models may show high accuracy. Still, they suffer from detecting classes in datasets that have few records. Therefore, in addition to accuracy, we used AUC as a measure to evaluate the ability of our classifier to distinguish between normal and malicious classes. Indeed, the ROC curve is used to assess performance of binary classification algorithms. Fig. 4 shows the ROC curve of the proposed model on the CICDDoS2019 dataset, which is very close to one. In fact, the AUC is 99.85% showing that the proposed model can distinguish benign and attack classes very well.

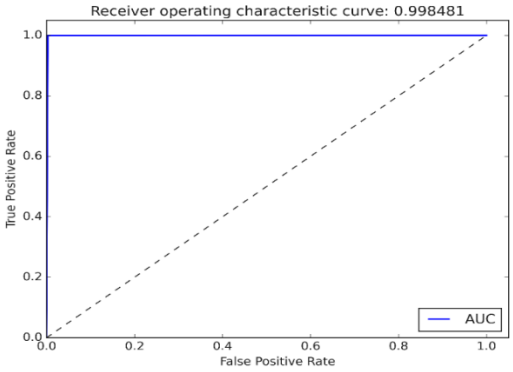

Fig. 4. ROC curve for our proposed method.

Table 4. Accuracy, AUC, and F1-score of MLP for DNS.

DrDoS_DNS	Accuracy	AUC	F1-score
GS	98.96	95.59	98.94
FS-GS	98.85	99.22	98.88

Table 5. Accuracy, AUC, and F1-score of MLP for NTP.

DrDoS_NTP	Accuracy	AUC	F1-score
GS	95.27	88.17	95.02
FS-GS	99.77	99.83	99.77

The set of MLP hyperparameters and their space which are given to the grid search algorithm as input are reported in Table 6.

Table 6. MLP hyperparameters space used in Grid Search.

Hyperparameter	Values
batch size	[1024, 2048, 4096]
hidden layer setting	[(83,83,11), (20,40,20), (15,15,15), (12,32,32,16), (10,18,26,10), (12,32,32,16,9), (10,18,26,12,6)]
activation	[sigmoid, tanh, relu, softmax]
solver	[sgd, adam]
alpha	[0.001, 0.01, 0.05, 0.1]
learning rate	[constant, adaptive]

The proposed model was compared with several methods, including the state-of-the-art work proposed in [26],

DOI: 10.22034/tjee.2022.15567

RNN-Autoencoder as a deep learning algorithm, and various traditional machine learning algorithms, such as random forest, decision tree, SVM, and logistic regression which are usually used in the literature as baseline [[36]]. According to the results presented in Table 7, our model consistently achieved better results in all metrics, compared to the RNN-Autoencoder algorithm and traditional machine learning algorithms. As expected, after our proposed model, the RNN-Autoencoder deep learning algorithm outperformed other machine learning algorithms in all criteria. Furthermore, the logistic regression algorithm performed better than SVM, random forest, and decision tree algorithms regarding accuracy, recall, and F1-score measures.

In addition to RNN and conventional machine learning methods, performance of the proposed MLP model was compared with a state-of-the-art work presented in [26]. The gradient boost (GB) [26] is a sequential ensemble learning technique used in classification and regression problems. The GB approach was analysed under four different cases in [26], for which, we report the average accuracy, AUC, precision, recall, and F1-score values that are obtained under the same circumstances as ours (imbalanced data, with hyperparameter tuning, and with feature selection). As it can be seen from Table 7, our MLP model outperforms GB in terms of all performance measures of interest.

Moreover, performance of the proposed MLP model is compared with the same MLP when the subset of CICDDoS2019 features selected in [26] are chosen. These features include protocol, total backward packets, total length of Fwd packets, total length of Bwd packets, flow IAT min, URG flag count, init Win bytes backward, inbound, and Bwd packet length Min. Then, the MLP model is trained and tested based on the above features. As it is shown in Table 7, our selected features provide better performance compared to the features used in [26]. Fig. 5 shows accuracy of the training procedure of the proposed approach on the CICDDoS2019 dataset during consecutive epochs. Here, the number of epochs is set to 200 but the algorithm is set to stop training if validation score is not improved in 5 successive iterations. As it can be seen in Fig. 5, the training phase terminates after 17 epochs.

Table 7. Performance of the proposed model in comparison with RNN-Autoencoder and four machine learning algorithms using the CICDDoS2019 dataset. [NA stands for not available].

Algorithm	Accuracy	AUC	Precision	Recall	F1-score
Decision Tree	77.00	NA	70.00	99.00	82.00
Random Forest	86.00	NA	100	74.00	85.00
SVM	93.00	NA	99.00	88.00	93.00
Logistic Regression	95.00	NA	93.00	99.00	96.00
RNN-Autoencoder	99.00	98.80	99.00	99.00	99.00
Gradient Boost [26]	99.89	97.11	97.56	94.26	95.79
Proposed MLP with features used in [26]	99.71	99.36	99.71	99.71	99.71
Proposed Model	99.92	99.85	99.92	99.92	99.92

6.1. Discussion

One of the main objectives of this research study is to demonstrate the potential of deep learning in anomaly detection systems. We applied a few techniques on the MLP as a deep learning model to accelerate classification of network inputs into normal and malicious classes. One of these techniques is dimensionality reduction. It refers to reducing the input variable using feature selection that allows our system to classify packets more quickly and accurately. Also, another technique which we used to find the algorithm's best parameters is the grid search. With implementing these techniques, the proposed model obtained outstanding results in terms of all metrics introduced in this paper, including recall, precision, F1-score, accuracy, and AUC compared to similar algorithms. The ability of deep learning to manage a high degree of complex nonlinear functions has made it one of the best techniques for detecting network intrusion. This capability allows deep learning to cope with the limitations of traditional classification methods. It also takes control and identifies the network offenders based on domain knowledge.

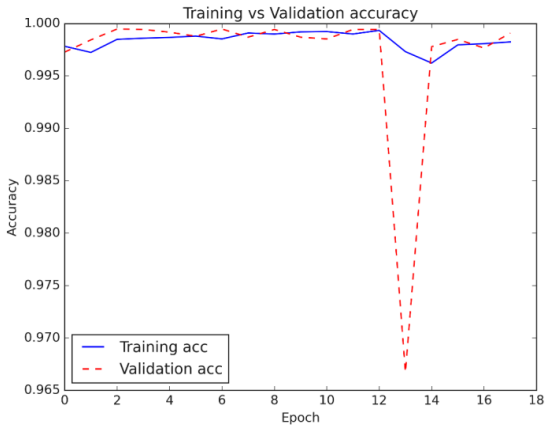

Fig. 5. Accuracy diagram of the proposed model in 10 epochs.

7. Sensitivity Analysis

Here, the sensitivity of output performance metrics to variations of input parameters of the proposed model is investigated. To do this, each input parameter is changed in a proper range, and then the sensitivity of the output to the variations of that parameter is studied. As discussed in earlier sections, several input parameters of the proposed model can affect the output measures of interest, including the hyperparameters indicated in Table 6. For brevity, we just report the sensitivity of accuracy, as a key metric, to variation of hyperparameter *alpha*. To achieve this, the value of alpha is changed from 0.001 to 0.1 with 0.05 steps. Fig. 6 shows the impact of variation of this hyperparameter on the accuracy of the proposed MLP model. As shown in Fig. 6, the accuracy of the model oscillates in an almost irregular way by changing alpha, though the best accuracy is achieved when alpha is 0.035. The behaviour is still in line with the output of the grid search algorithm where the value 0.001 for alpha outperforms 0.01, 0.05 and 0.1 values for that parameter in terms of accuracy. Comprehensive studies show that the sensitivity of other output metrics to the variation of alpha is almost the same.

DOI: 10.22034/tjee.2022.15567

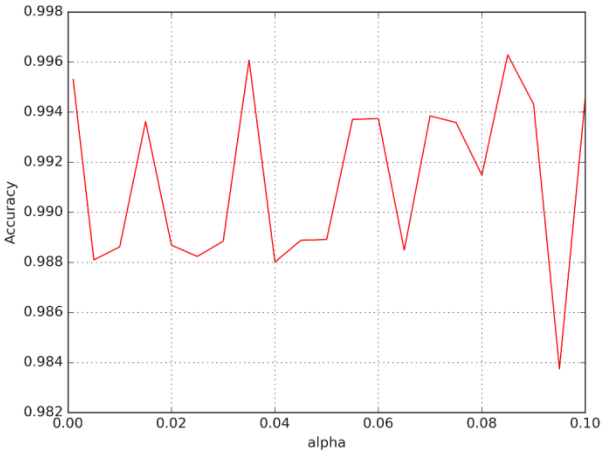

Fig. 6. Sensitivity of accuracy of the model to variation of alpha parameter.

8. Conclusion and Future Works

In this paper, a multi-layer perceptron model was implemented to detect DDoS attacks using the deep learning method. To increase the accuracy of the model, we obtained the best parameters of the model using the grid search algorithm. Feature selection was also used to reduce execution time, cost and detection errors. In this regard, we compared our proposed model with some state-of-the-art learning algorithms and the most frequently used and popular classical machine learning techniques as baselines. The proposed model was trained and tested using the CICDDoS2019 dataset, which encompasses the most recent DDoS attacks and other cyber threats. The results obtained from experimental investigations showed that our proposed model outperforms the baselines by 99.92% accuracy, and 99.85% AUC while reducing costs by eliminating less-valuable features.

In the future, we plan to test the proposed model with other types of attacks and cyber threats on the network. Also, we intend to measure effectiveness and efficacy of the presented model in a real environment under actual DDoS attacks.

9. References

[1] M. Ghasabi, M. Deypir, "Detection and mitigation of DDOS attacks in Software Defined Networks using the Jeffrey distance", *Tabriz Journal of Electrical Engineering*, vol. 48, pp. 1287–1300, 2018.

[2] IoT connected devices worldwide 2019-2030. In: Statista. https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/. Accessed 15 Nov 2021

[3] M.M. Salim, S. Rathore, J.H. Park, " Distributed denial of service attacks and its defenses in IoT: a survey", *Journal of Supercomputing*, vol. 76, pp. 5320–5363, 2020.

[4] I. Sharafaldin, A.H. Lashkari, S. Hakak, A.A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy", *In: 2019 International Carnahan Conference on Security Technology (ICCST), IEEE*, pp. 1–8, 2019.

[5] N.Z. Bawany, J.A. Shamsi, K. Salah, "DDoS attack detection and mitigation using SDN: methods, practices, and solutions", *Arabian Journal for Science and Engineering*, vol. 42, pp. 425–441, 2017.

[6] THESSLSTORE | The Largest DDoS Attacks in history. In: Hashed SSL Store™. https://www.thesslstore.com/blog/largest-ddos-attack-in-history/. Accessed 22 Dec 2020

[7] USENIX | The Advanced Computing Systems Association. https://www.usenix.org/. Accessed 23 Nov 2021

[8] Advancing IT, Audit, Governance, Risk, Privacy & Cybersecurity | ISACA. https://www.isaca.org/. Accessed 23 Nov 2021

[9] K. Singh, P. Singh, K. Kumar, "Application layer HTTP-GET flood DDoS attacks: Research landscape and challenges", *Computers & Security*, vol. 65, pp. 344–372, 2017.

[10] A.B. Dehkordi, M. Soltanaghaei, F.Z. Boroujeni, "The DDoS attacks detection through machine learning and statistical methods in SDN", *Journal of Supercomputing*, vol. 77, pp. 2383–2415, 2021.

[11] S. Behal, K. Kumar, "Detection of DDoS attacks and flash events using information theory metrics–an empirical investigation", *Computer Communications*, vol. 103, pp. 18–28, 2017.

[12] Y. Wang, "Analyses on limitations of information theory", *In: 2009 International Conference on Artificial Intelligence and Computational Intelligence, IEEE*, pp. 85–88, 2009.

[13] X. Yuan, C. Li, X. Li, "DeepDefense: identifying DDoS attack via deep learning", *In: 2017 IEEE International Conference on Smart Computing (SMARTCOMP), IEEE*, pp. 1–8, 2017.

[14] R. Doriguzzi-Corin, S. Millar, S. Scott-Hayward, "LUCID: A Practical, Lightweight Deep Learning Solution for DDoS Attack Detection", *IEEE Transactions on Network and Service Management*, 2020.

[15] F. Manavi, A. Hamzeh, "An Efficient Approach for Unknown Malware Detection Based on Opcode Analysis", *Tabriz Journal of Electrical Engineering*, vol. 50, pp. 1847–1864, 2021.

[16] M. Wang, Y. Lu, J. Qin, "A dynamic MLP-based DDoS attack detection method using feature selection and feedback", *Computers & Security*, vol. 88, pp. 101645, 2020.

[17] B. Shah, B.H. Trivedi, "Artificial neural network based intrusion detection system: A survey", *International Journal of Computer Applications*, vol. 39, pp. 13–18, 2012.

[18] R. Pradeepa, M. Pushpalatha, "IPR: Intelligent Proactive Routing model toward DDoS attack handling in SDN", *Journal of Supercomputing*, pp. 1–27, 2021.

[19] A. Saied, R.E. Overill, T. Radzik, "Detection of known and unknown DDoS attacks using Artificial Neural Networks", *Neurocomputing*, vol. 172, pp. 385–393, 2016.

[20] S. Sumathi, N. Karthikeyan, "Detection of distributed denial of service using deep learning neural

DOI: 10.22034/tjee.2022.15567

network", *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, pp. 5943–5953, 2021.

[21] Q. Niyaz, W. Sun, A.Y. Javaid, "A deep learning based DDoS detection system in software-defined networking (SDN)", *EAI Endorsed Transactions on Security and Safety ArXiv Preprint, ArXiv161107400*, 2016.

[22] R.M.A. Ujjan, Z. Pervez, K. Dahal, "Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN", *Future Generation Computer Systems*, vol. 111, pp. 763–779, 2020.

[23] K. Johnson Singh, K. Thongam, T. De, "Entropy-based application layer DDoS attack detection using artificial neural networks", *Entropy*, vol. 18, pp. 350, 2016.

[24] Z. He, T. Zhang, R.B. Lee, "Machine learning based DDoS attack detection from source side in cloud", *In: 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud). IEEE*, pp. 114–120, 2017.

[25] O.R. Sanchez, M. Repello, " Evaluating ML-based DDoS Detection with Grid Search Hyperparameter Optimization", *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, vol. , pp.402–408, 2021.

[26] R.K. Batchu, H. Seetha, "A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning", *Computer Networks*, vol. 200, pp.108498, 2021.

[27] M. Ismail, H. Hussain, A.A. Khan, U. Ullah, "A Machine Learning-Based Classification and Prediction Technique for DDoS Attacks", *IEEE Access*, vol. pp.21443–21454, 2022.

[28] A. Mihoub, O.B. Fredj, O. Cheikhrouhou, " Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques", *Computers & Electrical Engineering*, vol. 98, pp.107716, 2022.

[29] M. Alidoosti, A. Nowroozi, A. Nickabadi, "Assessing of Web Application Resiliency against Flooding DoS Attacks in the Business Layer", *Tabriz Journal of Electrical Engineering*, vol. 49, pp. 1757–1767, 2020.

[30] N.B. Gaikwad, V. Tiwari, A. Keskar, N.C. Shivaprakash, "Efficient FPGA implementation of multilayer perceptron for real-time human activity classification", *IEEE Access*, vol. 7, pp. 26696–26706, 2019.

[31] H.S. Das, P. Roy, "A deep dive into deep learning techniques for solving spoken language identification problems", *In: Intelligent Speech Signal Processing. Elsevier*, pp. 81–100, 2019.

[32] R. Atefinia, M. Ahmadi, "Network intrusion detection using multi-architectural modular deep neural network", *Journal of Supercomputing*, vol. 77, pp. 3571–3593, 2021.

[33] S. Ramírez-Gallego, B. Krawczyk, S. García, "A survey on data preprocessing for data stream mining: Current status and future directions", *Neurocomputing*, vol. 239, pp.39–57, 2017.

[34] J. Bergstra, Y. Bengio, "Random search for hyper-parameter optimization", *Journal of Machine Learning Research*, vol. 13(1), pp. 281-305, 2012.

[35] H.A. Fayed, A.F. Atiya, "Speed up grid-search for parameter selection of support vector machines", *Applied Soft Computing*, vol. 80, pp. 202–210, 2019.

[36] M.S. Elsayed, N.A. Le-Khac, S. Dev, A.D. Jurcut, "Ddosnet: A deep-learning model for detecting network attacks", *In: 2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM). IEEE*, pp. 391–396, 2020.

[37] C. Ferri, P. Flach, J. Hernández-Orallo, "Learning decision trees using the area under the ROC curve", *Conference: Machine Learning, Proceedings of the Nineteenth International Conference (ICML 2002)*, pp. 139–146, 2002.

[38] S.H. Park, J.M. Goo, C.H. Jo, "Receiver operating characteristic (ROC) curve: practical review for radiologists", *Korean Journal of Radiology*, vol. 5, pp. 11–18, 2004.

[39] M. Abadi, P. Barham, J. Chen, "Tensorflow: A system for large-scale machine learning", *In: 12th symposium on operating systems design and implementation*, pp. 265–283, 2016.

[40] Keras: the Python deep learning API. https://keras.io/. Accessed 13 Nov 2021.

[41] C.R. Harris, K.J. Millman, S.J. Walt, "Array programming with NumPy", *Nature*, vol. 585, pp. 357–362, 2020.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, "Scikit-learn: Machine learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[43] A. Géron, "Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems", *O'Reilly Media*, 2019.

[44] 1998 DARPA Intrusion Detection Evaluation Dataset | MIT Lincoln Laboratory. https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset. Accessed 15 Nov 2021.

[45] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory", *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, pp. 262–294, 2000.

[46] KDD Cup 1999 Data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Accessed 15 Nov 2021.

[47] (2010) The CAIDA "DDoS Attack 2007" Dataset. In: CAIDA. https://www.caida.org/catalog/datasets/ddos-20070804_dataset/. Accessed 15 Nov 2021.

[48] (2019) The CAIDA Anonymized Internet Traces Data Access. In: CAIDA. https://www.caida.org/catalog/datasets/passive_dataset_download/. Accessed 15 Nov 2021.

[49] M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set", *In: 2009 IEEE symposium on computational intelligence for security and defense applications, IEEE*, pp. 1–6, 2009.

DOI: 10.22034/tjee.2022.15567

[50] A. Shiravi, H. Shiravi, M. Tavallaee, A.A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection", *Computers & Security*, vol. 31, pp. 357–374, 2012.

[51] I. Sharafaldin, A.H. Lashkari, A.A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization", *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, vol. 1, pp.108–116, 2018.

[52] Applications | Research | Canadian Institute for Cybersecurity | UNB. https://www.unb.ca/cic/research/applications.html. Accessed 10 Nov 2021.

[53] DDoS 2019 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. https://www.unb.ca/cic/datasets/ddos-2019.html. Accessed 30 Nov 2021