IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 May 2023, accepted 23 May 2023, date of publication 25 May 2023, date of current version 1 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3280122

RESEARCH ARTICLE

Detection of Application-Layer DDoS Attacks Produced by Various Freely Accessible Toolkits Using Machine Learning

DYARI MOHAMMED SHARIF[1], HAKEM BEITOLLAHI[1,2], AND MAHDI FAZELI[3]

[1]Department of Computer Science, Faculty of Science, Soran University, Soran, Kurdistan Region 44008, Iraq
[2]School of Computer Engineering, Iran University of Science and Technology, Tehran 16846-13114, Iran
[3]School of Information Technology, Halmstad University, 301 18 Halmstad, Sweden

Corresponding author: Dyari Mohammed Sharif (dyari.ameen@gmail.com)

ABSTRACT Distributed Denial of Service (DDoS) attacks are a growing threat to online services, and various methods have been developed to detect them. However, past research has mainly focused on identifying attack patterns and types, without specifically addressing the role of freely available DDoS attack tools in the escalation of these attacks. This study aims to fill this gap by investigating the impact of the easy availability of DDoS attack tools on the frequency and severity of attacks. In this paper, a machine learning solution to detect DDoS attacks is proposed, which employs a feature selection technique to enhance its speed and efficiency, resulting in a substantial reduction in the feature subset. The provided evaluation metrics demonstrate that the model has a high accuracy level of 99.9%, a precision score of 96%, a recall score of 98%, and an F1 score of 97%. Moreover, the examination found that by utilizing a deliberate approach for feature selection, our model's efficacy was massively raised.

INDEX TERMS DDoS, DDoS tools, machine learning, deep learning.

I. INTRODUCTION

One of the most pernicious and increasingly complex security dangers to computer networks is distributed denial of service (DDoS) attacks [1], [2]. In Q1 2022, there was a significant increase in application-layer attacks, specifically HTTP-layer DDoS attacks which rose by 164% YoY and 135% QoQ. In terms of industry attacks, the Consumer Electronics sector experienced the highest increase with a staggering 5,086% QoQ. Online Media ranked second with a 2,131% increase in attacks QoQ, while Computer Software companies came in third with a 76% QoQ and 1,472 YoY increase in attacks [2].

A DDoS attack is a malevolent effort to stop a specific website, computer, or network from operating normally by saturating it with traffic from numerous sources. In this kind of attack, the perpetrator employs a network of computers or other devices (referred to as a "botnet") to overwhelm the target system with an excessive quantity of data, rendering it inaccessible to authorized users [4]. DDoS attacks are complex attacks since (1) they can generate a large volume of traffic from a wide variety of sources, and (2) the traffic appears to originate from a wide variety of locations [5].

DDoS attacks manifest in diverse forms, with application-layer attacks being one of them. Application-layer DDoS attacks target the application layer of the victim system, aiming to exhaust its resources or cause the application to fail. Illustrative examples of such attacks include HTTP floods and Slowloris attacks. The primary goal of application-layer DDoS attacks is to disable a network by overwhelming it with traffic, leading to system crashes or unavailability [6], [7], [8].

A significant factor in the growth of DDoS attacks is the easy availability of DDoS attack tools. These tools can be used intentionally to overwhelm servers and websites with traffic to the point where they become inoperable. Due to the

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau.

easy availability of tools, either through purchasing them on the dark web or downloading freely accessible scripts, people with little to no technical knowledge can carry out devastating DDoS attacks [4], [9], [10], [11].

In Literature, authors proposed various methods for detecting DDoS attacks; however, the majority of these studies focus on the identification of attack patterns and types rather than on the tools used to carry out the attacks. In contrast, our study places emphasis on the availability and accessibility of DDoS attack tools as a contributing factor to the growth of these attacks. Our aim is to analyze the impact of easy access to such tools on the frequency and severity of DDoS attacks, and to explore potential solutions for detecting this threat.

In order to address the problem of DDoS attacks brought on by the widespread accessibility of attack tools, we are proposing a machine learning (ML) solution. Specifically, we use Multi-layer perceptron (MLP) to identify the DDoS attacks. The solution aims to identify and distinguish between traffic produced by four freely accessible tools and legitimate traffic. To further enhance the system's speed and efficiency, a thorough feature selection technique, which is resulted in a significant reduction in the feature subset, is implemented. Moreover, based on our research findings, which have been compared with other relevant studies, it has been observed that training machine learning classifiers using specific attack tools leads to superior performance when defending against diversified DDoS attacks. The main contributions of our study is as follows:

- Proposed a ML solution to detect DDoS attacks by identifying and distinguishing between traffic produced by four freely accessible tools and legitimate traffic.
- Implemented a thorough feature selection technique to enhance the speed and efficiency of the system and reduce the feature subset from 78 to 6.
- Achieved high performance across multiple evaluation metrics, including 99.2% accuracy, 97.1% precision, 96.1% recall, and 96.6% F1 score when using Adam as the optimizer.
- Our main aim is to fill the gap by investigating the impact of the easy availability of DDoS attack tools on the frequency and severity of attacks. Past research has mainly focused on identifying attack patterns and types, without specifically addressing the role of freely available DDoS attack tools in the escalation of these attacks.

The rest of the paper is as follows: Sections II, III, IV, V and VI present the related work, DDoS launching tools, the proposed technique, Experiments and results and the conclusion, respectively.

II. RELATED WORK

A continual struggle in the field of cybersecurity has been the prevention of DoS/DDoS attacks. Researchers have presented a number of different ML-based approaches throughout the years to this problem. In this literature review, we examine a selection of previous efforts in the area.

Reference [12] employed supervised learning methods like Support Vector Machine (SVM) and Decision Tree (DT) C4.5 on NSL KDD Dataset for classification of DoS Attack. They use a sniffer to check up on the network's IP packets and identify malicious and benign ones. Evaluations reveal that accuracy acquired using DT C4.5 are more accurate than those obtained using SVM classifier.

Reference [13] suggests an approach for detecting DDoS attacks at the application layer that combines the Cuckoo Search Algorithm (CSA) and Radial Basis Function (RBF). The suggested approach uses a Genetic Algorithm (GA) for feature selection and a CSA to train an RBF neural network to detect DDoS, where the NSL-KDD dataset was used. In comparison to other techniques, it reliably detects application layer DDoS attacks with high performance.

Reference [14] suggested a method based on data preparation, feature selection, and classifiers that used rule-based classifiers. An information gain with ranker is employed to aid in the feature selection. The technique is built and validated using rule basis classifiers on dataset of GoldenEye tool in CICIDS2018. The decision table performed better than other rule-based algorithms.

Researchers in this publication [15] proposed a work on figuring out how to uncover the Low-Rate DoS (LDoS) attack's methodology and how it is generated. LDoS attacks are classified, and a filter protection strategy is identified to counter them. The main goal of the effort is to motivate academics to develop innovative techniques to detect and fight against LDoS attacks.

Reference [16] proposed that the Smith-Waterman local sequence alignment technique may be used to identify LDoS attacks by comparing the similarity scores of two different sequences. To properly estimate the distinctive parameters of the LDoS, this technique compares the locally generated detection sequence with the background flow, particularly the three characteristics of pulse period, duration, and amplitude of the constructed detection sequence. In order to determine the greatest possible score of similarity, they devise a two-threshold rule. Consequently, the goal of identifying a LDoS attack is met. As shown by the findings of the experiments, the suggested method is very accurate and has a high F1 score.

Reference [17] employed ML methods that are simulated using data collected from the switch. They implement DT C4.5, Artificial Neural Network (ANN) and K-nearest Neighbor (KNN). They identify DoS attacks that obtains a high detection rate and little false alarm.

A DoS attack detection system based on the Oppositional Crow Search Algorithm (OCSA), which combines the Crow Search Algorithm (CrSA) and Opposition Based Learning (OBL) approach, was proposed by [18]. System design includes two stages: feature selection using OCSA and classification with a Recurrent Neural Network classifier (RNN). After selecting the most important features using

the OCSA technique, they are sent to the RNN classifier. After that, the RNN classifier is used to sort the incoming data. In the experimental evaluation, the proposed technique outperforms other conventional methods in precision, recall, F-measure, and accuracy by 98.18%, 95.13%, and 93.56% and 94.12 %, respectively.

DoS detection using deep neural networks (DNN) was shown to be effective in every test case by [19]. An improved Deep Neural Network approach is presented for detecting DoS in this work. An adaptive particle swarm optimization technique selects the necessary parameters. Efficiency is determined by the ratio of packet transfer, energy consumption, delay and network length. A neural network with multiple layers of neurons is used to increase the detection precision while also cutting down on processing time. In the experiment, the results examines the effects of various optimization techniques on feature selection. Evaluations show that the new technique DNN has a greater detection ratio than the prior methods RAS-HO, TMS, and SVM-DoS.

Reference [20] investigates the performance of ML algorithms, including SVM, ANN, NB, DT, and unsupervised learning algorithm (USML), for Botnet DDoS attack detection using the UNBS-NB 15 and KDD99 datasets. The study reveals that USML is the most effective algorithm in accurately differentiating between Botnet and normal network traffic, with experimental results showing better performance for the KDD99 dataset

Reference [21] examines the security challenges posed by IoT, which is heterogeneous in nature and susceptible to various security threats such as DDoS attacks.The authors analyze existing DDoS variants, IoT security issues, the execution of DDoS attempts, and the creation of botnets or zombies from IoT devices.They also cover prevailing DDoS defense methodologies and their comparative analysis, as they provide a thorough understanding of DDoS over IoT and highlights the need for more intelligent and effective defense mechanisms to combat new variants of DDoS attacks and attacking methods.

Reference [22] is one of the works that is comparable to ours. The authors claimed to have successfully detected a DoS attack using ML and Neural Network (NN) approaches. The experiment utilizes the CIC IDS 2017 dataset as well as RF and MLP. When the results of RF and MLP are compared, it is clear that RF outperforms MLP. However, the proposed approach does not differentiate between attacks such as slowhttptest, slowloris, and http flood. In this work, we multi-classify the attacks. Furthermore, their study does not disclose any intelligible process for feature selection, but we devise an efficient mechanism for feature selection, and we are able to improve accuracy.

As we have reviewed the existing studies, we have found that although various techniques have been developed for detecting DDoS attacks, they primarily concentrate on recognizing the attack patterns and types rather than the tools used. Specifically, except for some limited number of studies, the focus of these research has been on identifying attack trends and categories rather than the freely available attack tools. Instead, we emphasize on how the easy availability of DDoS attack tools might affect the frequency and severity of attacks. Our study aims to determine and recommend efficient responses to this threat. More Specifically, our research differs from others in that we look closely at how publicly available DDoS attack tools have increased the number and intensity of these attacks. The main results from the research cited are outlined in Table 1.

III. DDoS LAUNCHING TOOLS

DDoS attacks may be conducted using a variety of tools, the majority of which are accessible to the general public online. High Orbit Ion Cannon (HOIC) and Low Orbit Ion Cannon (LOIC) are common DDoS attack tools for that. With the help of these tools, pervasive DDoS attacks can be carried out by attackers with limited technical expertise. Frequently, they provide a simple user interface that enables the attacker to define the target system, attack type, and attack intensity swiftly and simply [23]. Table 2 present the most prevalent DDoS attack launching tools:

In this work, we utilize a dataset where the traffic of HULK, GoldenEye, slowloris and slowhttptes are captured. The samples in the dataset are labeled based on the tool used to launch the attack. Both HULK and GoldenEye are DDoS attack tools capable of carrying out HTTP flood attacks, which include launching a large number of HTTP requests to a target server. A low-and-slow strategy is used by Slowloris and slowhttptest, two HTTP-based DDoS attack tools, to suck up server resources and prevent genuine users from accessing the service. In particular, Slowloris maintains connections open to the server, while slowhttptest makes HTTP GET and POST requests slowly to use up server resources [4].

IV. PROPOSED APPROACH

Most DDoS detection and mitigation methods to date have focused on specific attack types, such as HTTP floods, FTP floods, website spider attacks, asymmetric attacks, and slow header attacks. However, based on our extensive research in academia and industry, we have not found any evidence of researchers investigating the variety and accessibility of DDoS toolkits.

While those who possess a deep understanding of the attack environment can launch devastating attacks, there are also creative individuals who lack technical expertise but possess knowledge of toolkits capable of inflicting significant damage. When such toolkits are freely and widely available on the Internet and possess a high degree of variability, these individuals can become formidable attackers. They can select the optimal toolkit for a given target, time, and spot, and leverage the strengths of some attack toolkits to overcome the weaknesses of others and combine them, ultimately resulting in a devastating attack.

As it goes with the majority of models, our approach involves obtaining important data, cleansing the data, Data Normalization, determining the optimal feature set, and

TABLE 1. The summary of major findings from studies cited in the literature.

Study	Main Methods used	Data	Feature Selection	Key Findings	Evaluation Metrics used	No of features used	DDoS tools addressed
[12]	SVM & DT C4.5	NSL KDD	-	DT C4.5 is found to be more accurate than SVM	Accuracy	41	No
[13]	RBF & CSA	NSL KDD	GA	The suggested found to be more accurate than other existing techniques	Sensitivity & Specificity & Positive Predictive Value & Negative Predictive Value & Precision	9	No
[14]	Rule-based classifiers	GoldenEye in CI-CID2018	Information gain, Entropy with ranker	Decision table performs better than other rule-based algorithms	accuracy & precision & true positive rate & false alarm rate	23	Yes
[15]	Filter protection strategy	LDoS attacks	-	Effort is made to motivate the development of innovative techniques to detect and fight against LDoS attacks	Positive rate & False Positive Rate	-	No
[16]	Smith-Waterman local sequence alignment	LDoS attacks	Two-threshold rule	High accuracy and F1 score are achieved in detecting LDoS attacks	Recall & Precision & Accuracy & F1 score	-	No
[17]	DT C4.5, ANN, KNN	Data collected from the switch	-	High detection rate and low false alarms are achieved	Accuracy & Sensitivity & Specificity	6	No
[18]	OCSA & RNN	-	OCSA	Outperforms conventional methods in precision, recall, F-measure, and accuracy	Accuracy & Precision & Recall & F1 Score	10 & 12 & 15 & 16	No
[19]	Deep Neural Network	-	Adaptive particle swarm optimization	Greater detection ratio than prior methods RAS-HO, TMS, and SVM-DoS	-	3	No
[20]	SVM & ANN & NB & DT & USML	CICID 2017	-	The USML was the most effective in accurately differentiating between Botnet and normal network traffic for Botnet DDoS attack detection, as compared to other ML algorithms	Accuracy & False Alarm Rate & Sensitivity & Specifcity & False positive rate & AUC, and Matthews correlation coefcient	9-10	No
[21]	-	-	-	Identification of IoT device flaws that make them vulnerable to DDoS attacks and the need for more intelligent defense mechanisms to fight new DDoS types and attacking methods	-	-	Yes
[22]	RF & MLP	CICID 2017	-	RF outperforms MLP	Accuracy	80	No

classifying DoS/DDoS attacks. Figure 1 present the proposed approach. In this work, we propose a MLP to detect 5 classes of which 4 are DoS/DDoS attack types and one is the benign traffic. We start with a representative dataset, CICIDS2017 [24], which is publicly available for research community.

A. DATASET

The CICIDS2017 [24] dataset is used in this work where the abstract behavior of 25 users are profiled based on the HTTP, HTTPS, FTP, and SSH protocols, as well as email. The dataset originally has 692703 samples. HeartBleed attack samples are eliminated from the dataset, resulting in a final dataset of 692692 samples with having 78 features. Key properties and characteristics of the dataset:

- **Publicly available:** The dataset is publicly available, which makes it accessible to researchers and practitioners in the cybersecurity community. This promotes transparency and reproducibility in cybersecurity research and facilitates the development of more effective and

TABLE 2. The names of the DDoS launching tools and their corresponding definitions.

Tool Name	Definition
LOIC	It is a commonly used DoS/DDoS and open-source network stress testing tool among hackers and data breaches. It gives attackers the ability to inundate a target with traffic from several sources, such as TCP, UDP, and HTTP
HOIC	HOIC is a tool used for DoS/DDoS attacks and network stress testing. The TCP, UDP, and HTTP protocols are used in conjunction to saturate a target system with traffic
HULK	It is a Python-based program that can conduct HTTP flood attacks which a target server is bombarded with a large number of HTTP requests. It is intended to saturate the target server with traffic, blocking genuine users from accessing it.
GoldenEye	Another HTTP-based DDoS attack tool that is capable of doing HTTP flood attacks is called GoldenEye. It is similar to HULK in that it aims to overwhelm a target server with a large number of HTTP requests, but it also has some extra capabilities, such the option to utilize a unique user-agent header
Slowloris	A program developed expressly to attack web servers is called Slowloris. It employs a low-speed method to suck up server resources, blocking genuine users from accessing the service. It operates by opening as many connections as it can to the target server and maintaining them for as long as feasible
Slowhttptest	Another HTTP-based DDoS attack tool that utilizes a low-and-slow strategy to suck up server resources is slowhttptest. Slowly increasing the number of connections to the target server over time, it creates HTTP GET and POST requests

robust intrusion detection systems and machine learning models.

- **Realistic and diverse:** The dataset was generated in a realistic and diverse environment, which makes it a suitable representation of real-world network traffic. First, a comprehensive network setup was incorporated, including a modem, firewall, switches, and routers, as well as the existence of several operating systems such as Windows, Ubuntu, and Mac OS X. Second, the network's heterogeneity was taken into account since network traffic was captured from the primary switch, as well as memory dumps and system calls from all victim workstations, during the attack's execution.
- **Large and complex:** The dataset contains a large number of network flow records, with each record consisting of 78 different features. This makes it a complex dataset to work with and analyze. Since the mirror port, such as the tapping system, was employed, all traffics were collected and recorded on the storage server. Additionally, a Complete Interaction was also demonstrated as both within and between internal LANs were covered by having two distinct networks as well as Internet connectivity.
- **Labeled:** The dataset includes ground-truth labels for each network flow, indicating whether it is benign or malicious, and if malicious, which attack type it belongs to. This makes it a valuable resource for the development and evaluation of machine learning models and intrusion detection systems. Specifically, the data for each day was classified as "Benign" or "Attack." Each day, the sort of attack was different. The dataset document also included information on the timing of the attack.

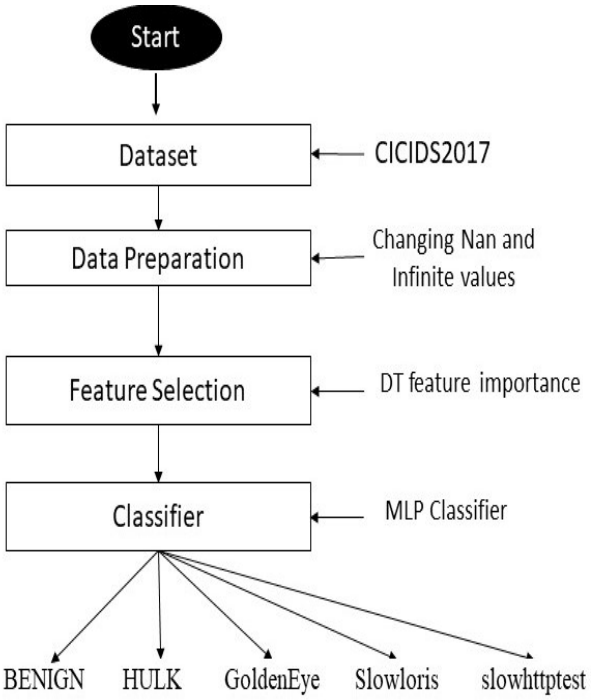

FIGURE 1. The overall structure of our proposed MLP.

TABLE 3. The class names and total number of observations within the dataset for each class.

Label	Number of records	In percentage
BENIGN	440031	63.52%
Attack: HULK	231073	33.36%
Attack: GoldenEye	10293	1.49%
Attack: slowloris	5796	0.84%
Attack: Slowhttptest	5499	0.79%

Table 3 present the summary of the names of the classes within the dataset and the total number of observations for each class.

We proceed by dividing the dataset into a training set consisting of 80% of the data and a testing set consisting of 20% of the data. Details regarding the dataset's train-test sets are presented in Table 4. The table summarizes the number and percentage of records for each class in both the training and testing sets

TABLE 4. The summarization of the train-test sets of the dataset including the quantity and proportion of entries for each type in both the training and testing sets.

Label	Training set		Testing set	
	Number of records	In percentage	Number of records	In percentage
BENIGN	351960	63.51%	88071	63.57%
Attack: Hulk	184855	33.36%	46218	33.36%
Attack: GoldenEye	8247	1.49%	2046	1.48%
Attack: slowloris	4682	0.84%	1114	0.80%
Attack: Slowhttptest	4409	0.80%	1090	0.79%

B. DATA PREPARATION

The process of preparing the data might be a time-consuming one, but it is essential in order to place the features in their proper perspective and eliminate bias from your model [25]. In this dataset, our data, with the exception of a few feature columns, was clean and ready to use. There are a total of 1008 nan values present inside the Flow Bytes/s feature. In addition, there are 289 and 1297 infinite values present inside the Flow Bytes/s and Flow Packets/s features, respectively. In this work, the nan and infinite values are substituted with the median and the maximum values of the feature column.

C. FEATURE SELECTION

When constructing a predictive model, one of the first steps is feature selection, which is the process of choosing the optimal feature subset. It is desired to limit the number of input features in order to increase the performance of the model and, in certain situations, to lower the computational cost of modeling [26]. In our work, we use DT for the purpose of feature selection. We populate all the data in the dataset to the DT then we consider the importance of features. The importance of a feature is determined as the (normalized) total reduction of the criteria that the feature contributes. It is also referred to as the Gini importance. After that, we take the mean of feature importance of all features and use it as a threshold. Any feature with feature importance below the threshold is discarded. The importance of each feature is then normalized and ranked, so that the most important feature has a score of 1, and the least important feature has a score of 0. The Feature importance in our work is calculated using the following formula, which is specific to the scikit-learn library for DTs:

$$\begin{aligned} \textit{Feature importance} = N_t/N * (\textit{impurity} - N_{t_R}/N_t \\ \times \ * \textit{right}_{\textit{impurity}} - N_{t_L}/N_t * \textit{left}_{\textit{impurity}}) \end{aligned} \tag{1}$$

where N is the total number of samples, N_t is the number of samples at the current node, N_{t_L} is the number of samples in the left child, and N_{t_R} is the number of samples in the right child. In the end of process, we are left with six features.

TABLE 5. The names and categories of the optimal subset of features.

Feature Name	Data Type
Destination Port	Integer
Bwd Packet Length Std	Float
Flow IAT Mean	Float
Bwd Packets/s	Float
Packet Length Mean	Float
Subflow Bwd Bytes	Integer

There are two integer and four float numbers. It is worht to note that we hold all prime essential features in order to detect diverse and various DDoS attack toolkits. The optimal feature subset is present in Table 5.

D. DATA SCALING

Data scalability is a guarantee of quality where Many ML approaches, such as gradient descent, the KNN algorithm, linear and logistic regression, and others, need it in order to provide useful results. Many different scalers have been created for this purpose [27]. This work uses the Min-Max Standard Scaler to transform the data into a number between the minimum and maximum values. With the help of Standard Scaler, a normal distribution with one standard deviation may be generated. It removes the mean value of a feature and divides the resulting value by the standard deviation of the feature in order to normalize it.

E. CLASSIFIER

Even though various methods for classifying data exist, it is hard to tell which one is the best without conducting extensive testing. The application and the type of data collecting are the deciding factors most of the time [28]. An artificial neural network is a collection of connected input/output units, each of which has a certain weight. During the learning phase, the network modifies the weights to better predict the correct class label of the input tuples [29]. Choosing the ideal number of neurons and layers in a MLP is an important step in building a neural network where several guidelines have been proposed in the literature. These include starting with a small number of hidden layers and increasing the number of layers only if necessary, selecting the number of neurons based on the complexity of the problem, using trial and error in combination with cross-validation techniques and regularizing the MLP to prevent overfitting [30], [31], [32]. It is therefore recommended that researchers adopt a systematic approach to this task to ensure the development of an effective neural network model. In this study, we use MLP to classify 4 DDoS attacks and benign traffic with 6 features only. To ensure a systematic approach to network classification, we adopt a stepwise strategy in selecting the appropriate number of neurons and hidden layers. Specifically, the procedure begins with no hidden layers, and considers the accuracy, precision, recall, and F1 score as performance metrics. The number of hidden layers is subsequently increased incrementally, with the corresponding number of neurons determined through

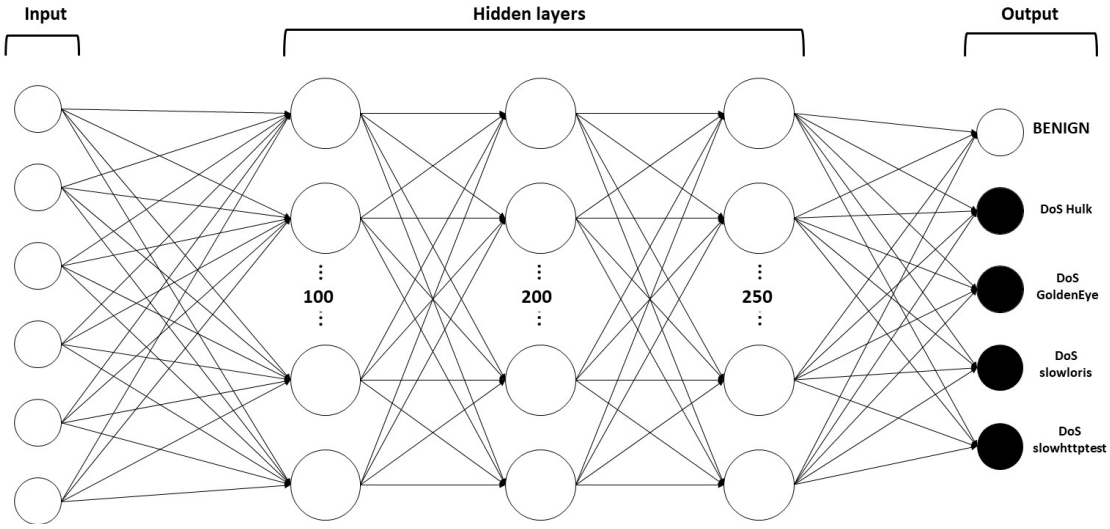

FIGURE 2. The overall structure of our proposed MLP.

Random Search (RS). The above process is repeated iteratively until a converged configuration is reached.

V. EXPERIMENTS AND RESULTS

All the practical aspects of our study are applied in Python. Our models are built using the Sklearn [33] library. After feature selection, the 6 selected features are populated to a MLP of three layers. The first hidden layer, second hidden layer and third hidden layer consists of 100,200 and 250 neurons, respectively. Figure 2 present the architecture of our proposed MLP.

When it came to deciding on features, we use Sklearn [33] to implement DT and then assesses the relevance of each feature importance by comparing it to the average of feature importance of all of them. The feature selection process goes this way: First, we consider the feature importance of each and every feature then we take the average of all of them and use it as a threshold. As a result, we manage to remove features whose feature importance falls below the threshold. Six features are chosen. Following that, the Max-Min Scaler is used to adjust the data such that it falls inside a certain range. The six features are populated to the MLP model with learning rate to 0.01, activation function being relu and random state to 0.01. We set tol=1e-4 and use a maximum of 50 iterations to input the six features into the MLP model. tol is the terminating criteria's tolerance which instructs scikit to terminate as soon as you're inside some tolerance, or as soon as you are near enough. Additionally, three different optimizers are used; Stochastic Gradient Descent (SGD), limited-memory BFGS (LBFGS), and Adam. For 10 consecutive epochs (iteration), training loss did not improve beyond tol=1e-4, it stopped at the iteration of 42. Based on the evaluation, our model obtains a level of accuracy of 99%, precision of 96%, a level of F1 score of 97%, and recall of 98% with adam optimizer. The results are shown in table 6 where Adam optimizer is utilized.

TABLE 6. The results of the proposed model using adam optimizer.

Metric	Result
Accuracy	99%
Precision	96%
Recall	98%
F1 Score	97%

There are a variety of optimizers to choose when developing ML methods. SGD, LBFGS, and Adam are three famous optimizers that are frequently used for MLPs, and are built-in in the Sklearn [33]. In order to develop deep learning models, stochastic gradient descent, a repetitive optimization method, is frequently used. Instead of calculating the gradient over the complete dataset, it does so for an arbitrarily chosen portion, and then uses that information to adjust the model's parameters. This can reduce the time spent exercising and help prevent overfitting [34].Comparatively, LBFGS is a quasi-Newton optimization method that estimates the loss function's Hessian matrix using second-order partial derivatives. It works better than SGD for some models, and it is often used to solve optimization problems that involve a lot of factors [35], [36].Finally, Adam, an extensively used optimization algorithm, blends SGD with momentum-based techniques. Faster convergence and improved generalization performance may result from its ability to adjust the learning rate for each parameter in response to the gradient and the gradient history [37], [38]. Moroever, it is important to note that the amount of the dataset, the intricacy of the model, and the intended performance metric all play a role in determining the optimal algorithm to use. In this work, we use all the three mentioned optimizers, finding the Adam the best optimizer for our case as proved in Table 7. Although accuracy is a vital

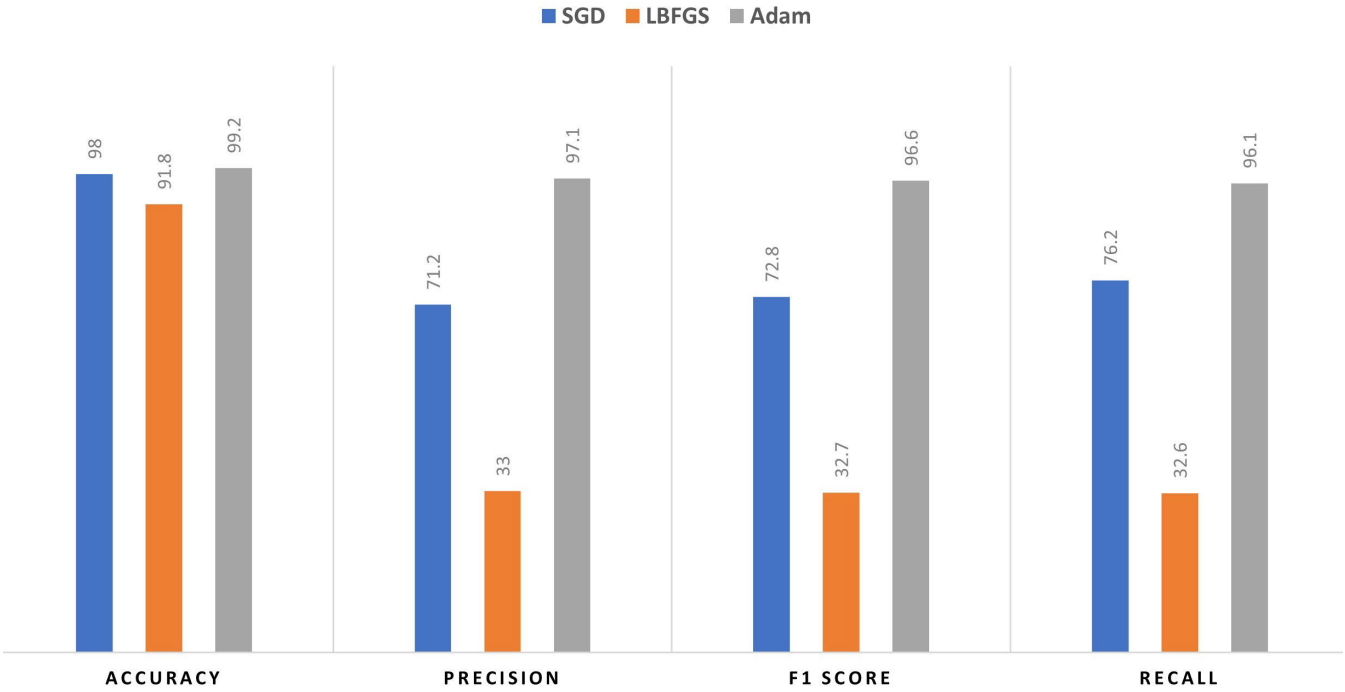

FIGURE 3. The results obtained using different optimizers.

TABLE 7. The comparison of using different optimizers for the MLP.

Metric	SGD	LBFGS	Adam
Accuracy	98.0%	91.8%	99.2%
Precision	71.2%	33.0%	97.1%
F1 Score	72.8%	32.7%	96.6%
Recall	76.2%	32.6%	96.1%

TABLE 8. The comparison of our method with previous related works.

	[20]	[18]	[22]	Proposed approach
Accuracy	90%	94%	96%	99%

statistic, other performance metrics such as precision, recall, and F1 score are also crucial to consider in order to get a full image of the model's efficacy. The results in an comparative manner are shown using the three optimizers in Table 7.

In terms of accuracy, Adam performs the best with a score of 99.2%, while SGD has the second-highest accuracy at 98.0%, and LBFGS has the lowest accuracy at 91.8%. For precision, Adam again has the highest score of 97.1%, followed by SGD with 71.2%, and LBFGS with the lowest precision score of 33.0%. The F1 score, which is a weighted average of precision and recall, also shows Adam as the top-performing optimizer with a score of 96.6%. SGD has an F1 score of 72.8%, while LBFGS has the lowest F1 score of 32.7%. Finally, in terms of recall, Adam has the highest score of 96.1%, followed by SGD with 76.2%, and LBFGS with the lowest score of 32.6%. Overall, the Adam outperforms others in all metrics utilized as are presented in Figure 3.

A. COMPARATIVE ANALYSIS OF THE PROPOSED METHOD WITH EXISTING WORKS

To have a fair comparison between the proposed method and previous state-of-the-art techniques, we train and test all techniques with the same dataset. It means that the same train part of the dataset is used to train all techniques and the same test part of the dataset is used to test all techniques. The proposed method is compared with unsupervised learning algorithm (USML) [20], the Oppositional Crow Search Algorithm (OCSA) plus RNN [18], and ML (machine learning) plus NN (neural network) [22]. Table 8 shows the comparison results in terms of accuracy. The proposed method provides better accuracy, as shown in the table.

VI. CONCLUSION

Distributed denial of service attacks (DDoS) has the potential to disable essential online services and prevent Users from accessing them. DDoS attack tools are of worry to the defense community for two reasons: (1) they are cheap or even free and easy to find online, and (2) hackers frequently employ them because they can be deployed with little in the way of technological expertise. According to our research, ML can be utilized to successfully classify DDoS tools into a total of five different categories; four of these classes pertain to DDoS tool attacks, and the remaining class contains innocuous data. By establishing a fast and reliable technique for selecting the features, we were able to cut the total number of options available in our model from 78 to 6. Our model has a high

level of performance across the board, with a 99.2% accuracy, 97.1% precision, 96.1% recall, and 96.6% F1 score when using Adam as the optimizer. These metrics show that our algorithm can correctly classify a significant amount of data despite its relatively small size. Our findings also highlight the significance of including only useful features in ML models. Our model's efficiency and our ability to understand the underlying data have both been boosted as a result of these changes. As a whole, our results show that our model is reliable and accurate at predicting the target variable; it has the potential to greatly enhance our capacity to detect and counteract DDoS attacks.

ACKNOWLEDGMENT

ChatGPT was used to refine the linguistic and writing capabilities when writing this paper. Its aid was helpful in terms of offering direction and support on syntax, structure, and terminology, which enhanced the end product's quality.

REFERENCES

[1] B. B. Gupta, P. Chaudhary, X. Chang, and N. Nedjah, "Smart defense against distributed denial of service attack in IoT networks using supervised learning classifiers," *Comput. Electr. Eng.*, vol. 98, Mar. 2022, Art. no. 107726.
[2] H. Beitollahi and G. Deconinck, "An overlay protection layer against denial-of-service attacks," in *Proc. IEEE Int. Symp. Parallel Distrib. Process.*, Apr. 2008, pp. 1–8.
[3] O. Yoachimik, "DDoS attack trends for 2022 Q1," Cloudflare, CA, USA, Tech. Rep., Apr. 2022.
[4] T. Shorey, D. Subbaiah, A. Goyal, A. Sakxena, and A. K. Mishra, "Performance comparison and analysis of Slowloris, GoldenEye and Xerxes DDoS attack tools," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2018, pp. 318–322.
[5] L. F. Eliyan and R. Di Pietro, "DoS and DDoS attacks in software defined networks: A survey of existing solutions and research challenges," *Future Gener. Comput. Syst.*, vol. 122, pp. 149–171, Sep. 2021.
[6] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 39–53, Apr. 2004.
[7] G. Somani, M. S. Gaur, D. Sanghi, M. Conti, and R. Buyya, "DDoS attacks in cloud computing: Issues, taxonomy, and future directions," *Comput. Commun.*, vol. 107, pp. 30–48, Jul. 2017.
[8] A. Bhardwaj, G. V. B. Subrahmanyam, V. Avasthi, H. Sastry, and S. Goundar, "DDoS attacks, new DDoS taxonomy and mitigation solutions—A survey," in *Proc. Int. Conf. Signal Process., Commun., Power Embedded Syst. (SCOPES)*, Oct. 2016, pp. 793–798.
[9] M. Sauter, "'LOIC will tear us apart' the impact of tool design and media portrayals in the success of activist DDOS attacks," *Amer. Behav. Scientist*, vol. 57, no. 7, pp. 983–1007, 2013.
[10] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, and D. Kumar, "Understanding the Mirai Botnet," in *Proc. 26th USENIX Secur. Symp.*, 2017, pp. 1093–1110.
[11] B. Nagpal, P. Sharma, N. Chauhan, and A. Panesar, "DDoS tools: Classification, analysis and comparison," in *Proc. 2nd Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, Mar. 2015, pp. 342–346.
[12] P. J. Shinde and M. Chatterjee, "A novel approach for classification and detection of DOS attacks," in *Proc. Int. Conf. Smart City Emerg. Technol. (ICSCET)*, Jan. 2018, pp. 1–6.
[13] H. Beitollahi, D. M. Sharif, and M. Fazeli, "Application layer DDoS attack detection using cuckoo search algorithm-trained radial basis function," *IEEE Access*, vol. 10, pp. 63844–63854, 2022.
[14] D. Kshirsagar and J. M. Shaikh, "Intrusion detection using rule-based machine learning algorithms," in *Proc. 5th Int. Conf. Comput., Commun., Control Autom. (ICCUBEA)*, Sep. 2019, pp. 1–4.
[15] W. Zhijun, L. Wenjing, L. Liang, and Y. Meng, "Low-rate DoS attacks, detection, defense, and challenges: A survey," *IEEE Access*, vol. 8, pp. 43920–43943, 2020.
[16] Z. Wu, Q. Pan, M. Yue, and L. Liu, "Sequence alignment detection of TCP-targeted synchronous low-rate DoS attacks," *Comput. Netw.*, vol. 152, pp. 64–77, Apr. 2019.
[17] O. Boyar, M. E. Özen, and B. Metin, "Detection of denial-of-service attacks with SNMP/RMON," in *Proc. IEEE 22nd Int. Conf. Intell. Eng. Syst. (INES)*, Jun. 2018, pp. 000437–000440.
[18] R. SaiSindhuTheja and G. K. Shyam, "An efficient metaheuristic algorithm based feature selection and recurrent neural network for DoS attack detection in cloud computing environment," *Appl. Soft Comput.*, vol. 100, Mar. 2021, Art. no. 106997.
[19] S. Ramesh, C. Yaashuwanth, K. Prathibanandhi, A. R. Basha, and T. Jayasankar, "An optimized deep neural network based DoS attack detection in wireless video sensor network," *J. Ambient Intell. Hum. Comput.*, pp. 1–14, 2021.
[20] T. A. Tuan, H. V. Long, L. H. Son, R. Kumar, I. Priyadarshini, and N. T. K. Son, "Performance evaluation of Botnet DDoS attack detection using machine learning," *Evol. Intell.*, vol. 13, no. 2, pp. 283–294, Jun. 2020.
[21] P. Kumari and A. K. Jain, "A comprehensive study of DDoS attacks over IoT network and their countermeasures," *Comput. Secur.*, vol. 127, Apr. 2023, Art. no. 103096.
[22] S. Wankhede and D. Kshirsagar, "DoS attack detection using machine learning and neural network," in *Proc. 4th Int. Conf. Comput. Commun. Control Autom. (ICCUBEA)*, Aug. 2018, pp. 1–5.
[23] Y. Jia, F. Zhong, A. Alrawais, B. Gong, and X. Cheng, "FlowGuard: An intelligent edge defense mechanism against IoT DDoS attacks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9552–9562, Oct. 2020.
[24] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, vol. 1, Jan. 2018, pp. 108–116.
[25] F. Ridzuan and W. M. N. Wan Zainon, "A review on data cleansing methods for big data," *Proc. Comput. Sci.*, vol. 161, pp. 731–738, Jan. 2019.
[26] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," *Neurocomputing*, vol. 300, pp. 70–79, Jul. 2018.
[27] M. Ahsan, M. Mahmud, P. Saha, K. Gupta, and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," *Technologies*, vol. 9, no. 3, p. 52, Jul. 2021.
[28] A. M. Mahfouz, D. Venugopal, and S. G. Shiva, "Comparative analysis of ML classifiers for network intrusion detection," in *Proc. 4th Int. Congr. Inf. Commun. Technol.* Cham, Switzerland: Springer, 2020, pp. 193–207.
[29] M. Al-Zewairi, S. Almajali, and A. Awajan, "Experimental evaluation of a multi-layer feed-forward artificial neural network classifier for network intrusion detection system," in *Proc. Int. Conf. New Trends Comput. Sci. (ICTCS)*, Oct. 2017, pp. 167–172.
[30] D. Hunter, H. Yu, M. S. Pukish, III, J. Kolbusz, and B. M. Wilamowski, "Selection of proper neural network sizes and architectures—A comparative study," *IEEE Trans. Ind. Informat.*, vol. 8, no. 2, pp. 228–240, May 2012.
[31] M. J. Madić and M. R. Radovanović, "Optimal selection of ANN training and architectural parameters using Taguchi method: A case study," *FME Trans.*, vol. 39, no. 2, pp. 79–86, 2011.
[32] G. Panchal, A. Ganatra, Y. P. Kosta, and D. Panchal, "Behaviour analysis of multilayer perceptronswith multiple hidden neurons and hidden layers," *Int. J. Comput. Theory Eng.*, pp. 332–337, 2011.
[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, no. 10, pp. 2825–2830, Jul. 2017.
[34] L. Bottou, "Stochastic gradient learning in neural networks," *Proc. Neuro-Nimes*, vol. 91, no. 8, p. 12, Nov. 1991.
[35] R. Wu, H. Huang, X. Qian, and T. Huang, "A L-BFGS based learning algorithm for complex-valued feedforward neural networks," *Neural Process. Lett.*, vol. 47, no. 3, pp. 1271–1284, Jun. 2018.
[36] A. S. Berahas and M. Takác, "A robust multi-batch L-BFGS method for machine learning," *Optim. Methods Softw.*, vol. 35, no. 1, pp. 191–219, Jan. 2020.
[37] I. K. M. Jais, A. R. Ismail, and S. Q. Nisa, "Adam optimization algorithm for wide and deep neural network," *Knowl. Eng. Data Sci.*, vol. 2, no. 1, pp. 41–46, 2019.
[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

DYARI MOHAMMED SHARIF received the Bachelor of Science degree in computer science/software from Soran University, Kurdistan Region, Iraq, in 2019, and the Master of Science degree in artificial intelligence, in 2022. His research interests include the development of DoS/DDoS detection and prevention techniques. He has been recognized for his academic achievements and contributions, having been awarded the Top Student Award from the Department Level, the Top Student at the Faculty Level, and the Top Ranked Research at the Department-Level.

HAKEM BEITOLLAHI received the B.S. degree in computer engineering from the University of Tehran, in 2002, the M.S. degree from the Sharif University of Technology, Tehran, Iran, in 2005, and the Ph.D. degree from Katholieke Universiteit Leuven, Belgium, in 2012. He is currently an Assistant Professor and the Head of the Hardware and Computer Systems Architecture Branch, School of Computer Engineering, Iran University of Science and Technology. His research interests include network security, real-time systems, fault tolerance, and dependability.

MAHDI FAZELI received the M.Sc. and Ph.D. degrees in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2005 and 2011, respectively. He is currently an Associate Professor with the School of Information Technology, Halmstad University, Sweden. He has authored or coauthored more than 50 papers in reputable journals and conferences. His current research interests include system level power analysis and management, real-time systems, hardware security, and reliability modeling and evaluation.