Detection of Distributed Denial of Service Attacks using Automatic Feature Selection with Enhancement for Imbalance Dataset

Duy-Cat Can[1], Hoang-Quynh Le[1], and Quang-Thuy Ha[1]

Faculty of Information Technology, University of Engineering and Technology
Vietnam National University Hanoi, Vietnam
{catcd,lhquynh,thuyhq}@vnu.edu.vn

Abstract. With the development of technology, the highly accessible internet service is the biggest demand for most people. Online network, however, has been suffering from malicious attempts to disrupt essential web technologies, resulting in service failures. In this work, we introduced a model to detect and classify Distributed Denial of Service attacks based on neural networks that take advantage of a proposed automatic feature selection component. The experimental results on CIC-DDoS 2019 dataset have demonstrated that our proposed model outperformed other machine learning-based model by large margin. We also investigated the effectiveness of weighted loss and hinge loss on handling the class imbalance problem.

Keywords: DDoS attack · Multi-layer Perceptron · Automatic feature selection · Class imbalance · Multi-hinge loss · Weighted loss.

1 Introduction

A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt regular traffic of a targeted server, service, or network by overwhelming the target or its surrounding infrastructure with a flood of traffic from illegitimate users [15]. It aims at depleting network bandwidth or exhausting target's resources with malicious traffic. This attack causes the denial of service to authenticated users and causes great harm to the security of network environment.

A DDOS attack is relatively simple but often brings a disturbing effect to Internet resources. Together with the popularity and low-cost of the Internet, DDoS attacks have become a severe Internet security threat that challenging the accessibility of resources to authorized clients. According to the forecast of the Cisco Visual Networking Index, the number of DDoS attacks grew 172% in 2016, and expects that this will increase 2.5-fold to 3.1 million by 2021 globally[1].

Basically, DDoS attacks are based on the same techniques as another regular denial of service (DoS) attacks. The differences are, *(i)* it uses a single network

[1] https://www.infosecurity-magazine.com/news/cisco-vni-ddos-attacks-increase/, archived on 11 November, 2020

connection to flood a target with malicious traffic, and (ii) it uses botnets to perform the attack on a much larger scale than regular DOS attacks [4]. A botnet is a combination of numerous remotely managed compromised hosts (i.e., bots, zombies, or other types of slave agents), often distributed globally. They are under the control of one or more intruders. This work focuses on attack detection, that identifying the attacks immediately after they actually happen to attack a particular victim with different types of packets. The experts define several kinds of DDoS attacks; examples include UDP Flood, ICMP (Ping) Flood, SYN Flood, Ping of Death, Slowloris, HTTP Flood, and NTP Amplification [13].

DDoS defense system consists of four phases: Attack prevention, Attack detection and characterization, Traceback and Attack reaction [4]. This work focuses on attack detection, that identifying the attacks immediately after they actually happen. In the case of a system is under DDoS attack, there are unusual fluctuations in the network traffic. The attack detector must automatically monitor and analyze these abrupt changes in the network to notice unexpected problems [8]. In this work, We consider this problem as a classification problem, i.e., classifies DDoS attacks packets and legitimate packets.

Although many statistical methods have been designed for DDoS attack detection, designing an effective detector with the ability to adapt to change of DDoS attacks automatically is still challenging so far. This paper proposes a deep learning-based model for DDoS detection that selects the features automatically. The experiments were conducted on CIC-DDoS 2019 dataset [20]. Evaluation results show that our proposed model achieved significant performance improvement in terms of $F1$ compared to the baseline model and several existing DDoS attacks detection methods.

The main contributions of our work can be concluded as:

i. We represent a deep neural network model to detect DDoS attacks that utilize many improvement techniques.
ii. We propose and demonstrate the effectiveness of automatic feature selection method.
iii. We investigate the contributions of multi-hinge loss and weighted loss to handling class imbalance problem.

2 Proposed Model

Figure 1 depicts the overall architecture of our proposed model DDoSNet. DDoSNet mainly consists of two components: a feature selection layer, and a classification layer using fully-connect multi-layer perceptron (MLP). Given a set of traffic features as input, we build an automatic feature selection model to calculate a weight for each feature. An MLP model is applied to capture higher abstract features, and a softmax layer is followed to perform a (K+1)-class distribution. The details of each layer are described below.

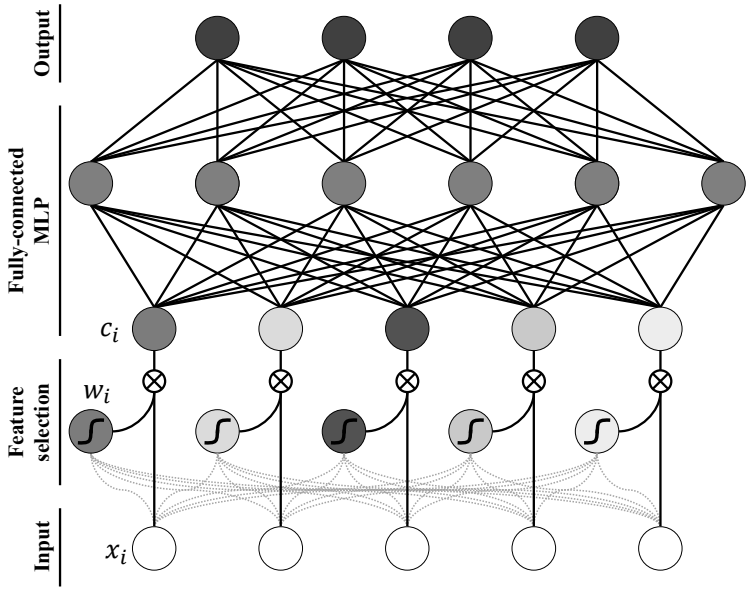


Fig. 1: An overview of proposed model.

2.1 Data preprocessing

In the first step of implementation, the preprocessing on our datasets is exerted. We follow four preprocessing operations to prepare the data before the module training.

- **Removal of irrelevant features**: we remove all of the attributes which are non informative such as `Unnamed: 0`, `Flow ID`, `Timestamp`, and all of the socket features like `Source IP`, `Destination IP`.
- **Cleaning the data**: we have convert invalid values such as `NaN` and `inf` or `SimillarHTTP` to corresponding value for efficient running of algorithms.
- **Label Encoding**: One-hot encoding is used to convert categorical label into numerals. In addition, we use another binary label 1 and 0 to denote if an example is an DDoS attack or not.
- **Normalization**: the features data have different numerical range value that make training process biasing on large values. For the random features, we normalize these features to normal distribution, as follow:

$$x_i = \frac{f_i - \mu_i}{\sigma_i} \tag{1}$$

where μ and σ are feature mean and standard deviation respectively. For the fixed value features, we normalize the data using min-max scaling as follow:

$$x_i = \frac{f_i - f_{min}}{f_{max} - f_{min}} \tag{2}$$

2.2 Feature Selection

Feature selection is one of the key problems for machine learning and data mining that selecting a feature subset that performs the best under a certain evaluation criterion. Sharafaldin et al. [20] have used Random Forest Regressor to examine the performance of the selected features and have selected 24 best features with corresponding weight for each DDoS attack.

In this paper, we proposed to use a simple neural network to select and learn important weights for input feature set. Given a feature set of n features, for each input feature x_i, we calculate the context attention score base on the whole feature set and the value of feature itself. The attention score s_i that have taken into account the feature set context is then transformed into a weight in range $[0, 1]$. Finally, the feature weight is multiplied to the corespondent feature value. This procedure is described in formula given below:

$$\mathbf{h}_i = \tanh(\mathbf{x}\mathbf{W}^x + [x_i]\mathbf{W}^{x'} + \mathbf{b}^h) \tag{3}$$

$$s_i = \mathbf{h}_i\mathbf{w}^s + b^s \tag{4}$$

$$w_i = \frac{1}{1 + e^{-s_i}} \tag{5}$$

$$c_i = x_i w_i \tag{6}$$

where $\mathbf{W}^x \in \mathbb{R}^{n\times h}$, $\mathbf{W}^{x'} \in \mathbb{R}^{1\times h}$, $\mathbf{b}^h \in \mathbb{R}^h$ are weights and bias for hidden attention layer; $\mathbf{w}^e \in \mathbb{R}^h$ and $b^e \in \mathbb{R}$ are weights and bias for attention score.

2.3 Classification

The features from the penultimate layer are then fed into a fully connected multi-layer perceptron network (MLP). We choose hyperbolic tangent as the non-linear activation function in the hidden layer. I.e.

$$\mathbf{h} = \tanh\left(\mathbf{c}\mathbf{W}_h + \mathbf{b}_h\right) \tag{7}$$

where $\mathbf{W}_h$ and $\mathbf{b}_h$ are weight and bias terms. We apply multi hidden layer to produce higher abstraction-level features The output h of the last hidden layer is the highly abstract representation of input features, which is then fed to a softmax classifier to predict a (K+1)-class distribution over labels $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \text{softmax}\left(\mathbf{h}\mathbf{W}_y + \mathbf{b}_y\right) \tag{8}$$

2.4 Objective Function and Learning Method

We compute the the penalized cross-entropy, and further define the training objective for a data sample as:

$$L(\hat{\mathbf{y}}) = -\sum_{i=0}^{K} \mathbf{y}_i \log \hat{\mathbf{y}}_i \tag{9}$$

where $\mathbf{y} \in \{0,1\}^{(K+1)}$ indicating the one-hot vector represented the target label. In addition to categorical cross entropy losses, we use hinge loss to generate a decision boundary between classes. We use the formula defined for a linear classifier by Crammer and Singer [3] as follow:

$$\ell(y) = \max(0, 1 + \max_{y \neq t} \mathbf{x}\mathbf{w}_y - \mathbf{x}\mathbf{w}_t) \tag{10}$$

where t is the target label, $\mathbf{w}_t$ and $\mathbf{w}_y$ are the model parameters. We further add L^1-norm and L^2-norm of model's weights and L^2-norm of model's biases to model objective function to keep parameter in track and accelerate model training speed.

$$L(\theta) = \alpha \left\|\mathbf{W}\right\|_2 + \beta \left\|\mathbf{W}\right\|_1 + \lambda \left\|\mathbf{b}\right\|_2 \tag{11}$$

where α, β and λ are regularization factors.

The model parameters $\mathbf{W}$ and $\mathbf{b}$ are initialized using Xavier normal initializer [7] that draws samples from a truncated normal distribution centered on 0. To compute these model parameters, we minimize $L(\theta)$ by applying Stochastic Gradient Descent (SGD) with Adam optimizer [14] in our experiments.

To handle the class imbalance problem, we drive our model to have the classifier heavily weight the few examples that are available by using weighted loss. We calculate the weight for each class as follow:

$$w_i = \frac{1}{n_i} \frac{1}{2} \sum_j n_j \tag{12}$$

where n_i is number of class i examples. Scaling by total/2 helps keep the loss to a similar magnitude. The sum of the weights of all examples stays the same.

3 Experiment and Discussion

3.1 CIC-DDoS 2019 Dataset

Many data sets are using in studies that are made using different algorithms in Intrusion Detection System designs. In this paper, we evaluate our proposed classifier using the new released CIC-DDoS 2019 dataset [20] which was shared by Canadian Institute for Cybersecurity.The dataset contains a large amount of different DDoS attacks that can be carried out through application layer protocols using TCP/UDP.

Table 1 summarizes the distribution of the different attacks in the CIC-DDoS 2019 dataset. The dataset was collected in two separated days for training and testing evaluation. The training set was captured on January 12th, 2019, and contains 12 different kinds of DDoS attacks, each attack type in a separated PCAP file. The attack types in the training day includes DNS, LDAP, MSSQL, NetBIOS, NTP, SNMP, SSDP, Syn, TFTP, UDP, UDPLag, and WebDDoS DDoS

Table 1: Statistic of training and testing dataset.

Label	Training dataset		Testing dataset	
	Num.	Per	Num.	Per
BENIGN	56863	0.11%	56965	0.28%
DNS	5071011	10.13%	-	-
LDAP	2179930	4.35%	1915122	9.40%
MSSQL	4522492	9.03%	5787453	28.42%
NetBIOS	4093279	8.18%	3657497	17.96%
NTP	1202642	2.40%	-	-
Portmap	-	-	186960	0.92%
SNMP	5159870	10.31%	-	-
SSDP	2610611	5.21%	-	-
Syn	1582289	3.16%	4891500	24.02%
TFTP	20082580	40.11%	-	-
UDP	3134645	6.26%	3867155	18.99%
UDPLag	366461	0.73%	1873	0.01%
WebDDoS	439	0.00%	-	-
Total	**50063112**	**100%**	**20364525**	**100%**

based attack. The testing data was created on March 11th, 2019, and contains 7 DDoS attacks LDAP, MSSQL, NetBIOS, Portmap, Syn, UDP, and UDPLag.

The training and testing datasets vary in distribution of data. For example, two minor classes MSSQL and NetBIOS in training dataset are major class in testing dataset with percentage of 28.42% and 17.96% respectively. The class imbalance is also a challenge of this dataset in which minor classes account for less than 1%. Another notable remark is more than 68% of training dataset belong to the classes are totally absent from testing dataset. The Portmap attack in the testing set dose not present in the training data for intrinsic evaluation of detection system.

Experimental configuration: In the experiments, we fine-tune our model on 90% of training dataset and report the results on the testing dataset, which is kept secret with the model. We leave 10% of training dataset for validation dataset to fine-tune model's hyper-parameters. We conduct the training and testing process 10 times and calculate the averaged results. For evaluation, the predicted labels were compared to the golden annotated data with common machine learning evaluation metrics: precision (P), recall (R), and F1 score.

3.2 System's performance

We compared our model with various common machine learning algorithms namely decision tree, random forest, Naïve Bayes and logistic regression that reported by Sharafaldin et al. [20]. These performance examination results are in terms of the weighted average of the evaluation metrics with five-fold cross validation. For a fair comparison, we re-implemented these models and evaluate

Table 2: System's performance on CIC-DDoS 2019 dataset.

Model		Average			Binary		
		P	**R**	**F1**	**P**	**R**	**F1**
Benchmark [20]†	Decision tree	78.00	65.00	69.00	-	-	-
	Random forest	77.00	56.00	62.00	-	-	-
	Naïve Bayes	41.00	11.00	5.00	-	-	-
	Logistic regression	25.00	2.00	4.00	-	-	-
Elsayed et al. [5]	Random forest	-	-	-	100.00	74.00	85.00
	SVM	-	-	-	99.00	88.00	93.00
	Logistic regression	-	-	-	93.00	99.00	96.00
	RNN-Autoencoder	-	-	-	99.00	99.00	99.00
Baseline‡	Naïve Bayes	30.30	17.51	7.35	-	-	-
	SVM	62.44	57.97	55.50	-	-	-
	Decision Tree	61.15	58.32	55.15	-	-	-
	Random Forest	50.76	36.91	39.57	-	-	-
Our model‡	24 features	91.12	72.91	74.00	99.99	99.93	99.96
	24 features + FS	85.19	76.51	75.44	99.99	99.89	99.94
	82 features	88.97	70.61	71.09	99.99	99.94	99.96
	82 features + FS	**91.16**	**79.41**	**79.39**	**99.98**	**99.89**	**99.93**
	- hinge loss	82.06	73.60	74.29	99.99	99.93	99.96
	- weighted loss	60.51	67.34	63.60	99.97	99.94	99.95

† *5-fold cross validation, weighted average*

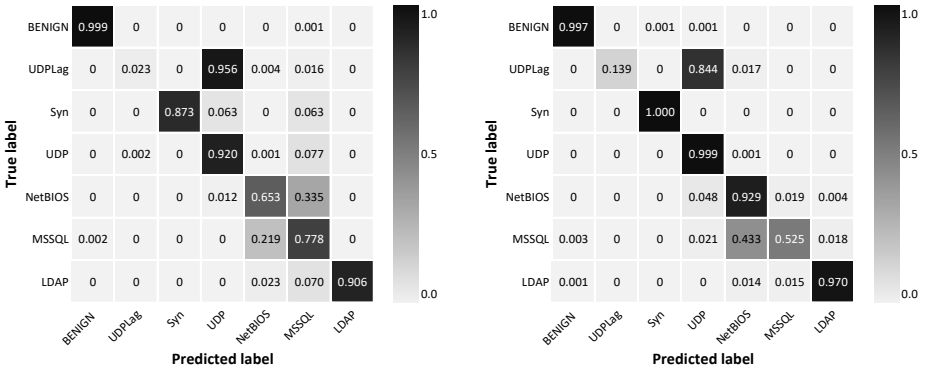
‡ *train-test split, macro average*

on separated training and testing datasets. Table 2 represents the classification metrics of our six model variants with different comparative models.

According to benchmark results, decision tree (ID3) performed the best with the fastest training time.Random forest is follow with the result of 69% on more than 15 hours of training. The Naïve Bayes classifier performed poorly, primarily because the NB assumed that all attributes are independent of each other. Finally, logistic regression, with more than 2 days of training process, did not meet the expectation with 5% F1 score.

Our reproduced baseline results on separated training and testing data have similarities with the benchmark results. Decision tree gives high performance at 55.15% F1, followed by random forest and Naïve Bayes with 39.57% and 7.35% respectively. In addition, we also try applying support vector machine (SVM) and have slightly better results than other methods.

The obtained results show that our model outperforms the other machine learning algorithms by large margin. Firstly, we apply our deep learning model directly on the input examples without automatic feature selection component. We observe that our proposed model produces better result on 24 selected features introduced in study of Sharafaldin et al. [20] with 2.91% gap with the model applied on all 82 features. However, when applied the automatic feature selection based on feature context vector, we notice the complete opposite results. With the improvement of 8.3%, 82-feature model (DDoSNet) yield the best



(a) Without multi-hinge loss. (b) With multi-hinge loss.

Fig. 2: Model's prediction confusion matrix.

F1 result. Meanwhile, the 24-feature model showed only a small improvement of about 1.44%. One possible reason is the feature weights are calculated based on the feature context vector, 82 features, therefor, give more information.

We also considered the binary result and compared our model with another neural network-based model (RNN-Autoencoder) that proposed by Elsayed et al. [5]. In this experiment, we have witnessed the dominance of deep learning models. The logistic regression model that gave poor results with imbalanced multi-class data has been re-vital that gave high results with binary data. RNN-autoencoder model as well as the our proposed deep learning models performed excelent on this binary data with over 99% of F1. Deep learning models rarely misclassified which example is DDoS attack.

3.3 Result analysis

Class imbalance problem: CIC-DDoS 2019 is an imbalanced dataset, in which 2 major classes account for over 50%, the ratio between the largest and smallest class in the test set is more than 3000 times. We have done some further investigations into the experimental results. Figure 2b presents the confusion matrix of DDoSNet model's prediction on validation dataset. As we observe on the confusion matrix, examples of the BENIGN class - not a DDoS attack - are rarely confused with attack classes and vice versa. Among the attack classes, the syn and LDAP classes also performed well without being misclassified with the other classes. In contrast, 84.8% of inputs from UDPLag class were mistakenly classified as a UDP attack, causing the recall metric of UDPLag to drop to 11.28%. This can be explained by two reasons: (i) UDPLag is a minor label - the percentages in training and testing set are only 0.73% and 0.01% respectively - so classifiers are difficult to recognize the data belongs to this class; (ii) on the DDoS attack taxonomy tree, UDPLag is a child-node of UDP so UDPLag

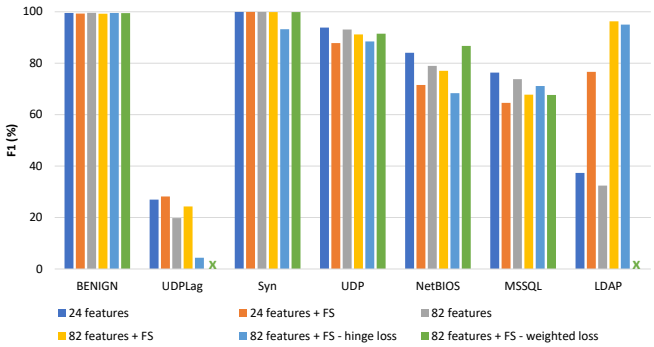


Fig. 3: Comparison of each label's F1 score of 6 model variants. X columns denote 0.0% of F1 score.

examples collapsed into UDP class is reasonable. Another class did not perform well is MSSQL with 43.3% of the input being mistaken to NetBIOS.

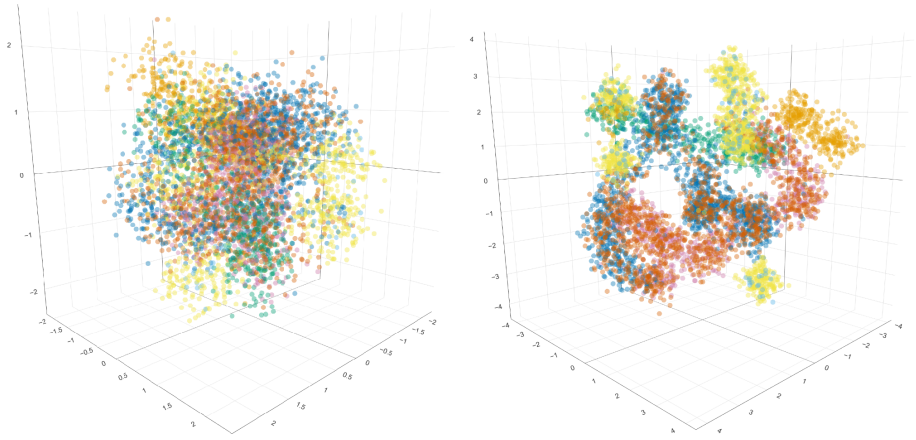
Another analysis on the results of each classes with different variations of the proposed model is summarized in Figure 3. According to the statistics of model variants' results, class weight plays an important role in training the model to predict minor classes. When removing the weighted loss, the results of two classes UDPLag and LDAP dropped to 0.0%. The automatic feature selection component also plays a certain role in solving the class imbalance problem, the most obvious demonstration shown in the LADP class result. Another interesting observation is that although Syn was a minor class in the training set, the test results of this label exceeded our expectation. One possible reason is Syn label is on a separate branch on the taxonomy tree, so the features of this class are obvious making machine learning models easy to detect.

Experiment of Automatic Feature Selection: In this experiment, to analyze the efficiency of automatic feature selection module, we re-executed our model on 100,000 random validation examples and extracted the weight for each feature. The arithmetic mean of the weights of each feature by classes is presented in Figure 4. Observing the weighted heat-map of input features, we have seen that the important levels that our model learned for BENIGN label is often in opposition to attack labels. The `ACK Flag Count`, `Destination Port`, `Init_Win_bytes_forward`, `min_seg_size_forward` and `protocol` features have been highlighted as the most important features for distinguishing types of DDoS attacks. When compared with the weights that have been meticulously selected through experiments in the study of Saharafaldin et al. [20], our automatically selected features have a lot of similarities. However, some of our weights are in stark contrast to the above study. For example, `Flow IAT Min` and `Fwd Packet Length Std` for BENIGN class, `ACK Flag Count` and `Fwd IAT Total` for Syn class, and `Average Packet Size` and Fwd `Packet Length Max` for LADP class.

	BENIGN	UDPLag	Syn	UDP	NetBIOS	MSSQL	LDAP
ACK Flag Count	**0.9194**	**1.0000**	**0.0145**	0.4956	0.5243	0.5658	0.5741
Average Packet Size	0.9938	0.0000	0.0764	0.0000	0.0005	0.0036	**0.0009**
Destination Port	0.0062	0.6214	0.9382	**1.0000**	0.3798	0.6173	0.5148
Flow Duration	0.0062	0.5235	**0.9382**	0.7849	0.5794	0.4563	0.5089
Flow IAT Max	0.0016	0.4878	0.0000	0.5848	0.4339	0.4963	0.2997
Flow IAT Mean	0.0062	0.4714	0.5462	0.4112	0.3913	0.5299	0.4268
Flow IAT Min	**0.0042**	0.5291	0.5048	0.4419	0.6048	0.6311	0.6151
Fwd Header Length	0.0016	0.4171	0.0050	0.5338	**0.9995**	0.4002	0.5097
Fwd Header Length.1	0.0016	0.4793	0.0000	0.4069	**0.9995**	0.4347	0.4856
Fwd IAT Max	0.7699	**0.8352**	0.3418	0.1108	0.1929	0.2244	0.1947
Fwd IAT Mean	0.0016	**0.9731**	0.0145	0.5085	0.4264	0.3819	0.3716
Fwd IAT Total	0.0016	0.4807	**0.0189**	0.4975	0.3946	0.5392	0.5523
Fwd Packet Length Max	0.0307	0.3542	0.5488	0.5197	0.5232	0.4901	**0.0983**
Fwd Packet Length Min	0.2444	0.7643	0.4765	0.7650	0.7779	0.7917	**0.8063**
Fwd Packet Length Std	**0.0060**	0.3566	0.8841	**1.0000**	0.6157	0.6241	0.4528
Fwd Packets/s	0.5628	0.4382	0.3884	0.4369	**0.4783**	**0.4914**	0.5018
Init_Win_bytes_forward	**0.0062**	**1.0000**	**0.9361**	0.5547	0.3279	0.5133	0.5226
Max Packet Length	0.4945	0.4986	0.7448	0.4997	0.5094	0.5020	**0.4957**
Min Packet Length	0.0068	0.4994	0.4899	0.4698	0.5684	0.4081	**0.9991**
min_seg_size_forward	0.0040	**1.0000**	**0.4880**	**1.0000**	**0.9995**	0.6211	0.6160
Packet Length Std	**0.4066**	0.6011	0.1491	**0.6021**	0.6234	0.6462	0.6668
Protocol	0.0062	0.4678	0.5121	**1.0000**	**0.9995**	**0.9964**	0.5558
Subflow Fwd Bytes	0.9938	0.0000	0.0619	0.0000	0.0005	0.0036	0.0009
Total Length of Fwd Packets	0.0061	0.4225	0.4380	0.4015	0.7212	0.5502	0.3591

0.0 1.0

Fig. 4: Weight of 24 features corresponding to each label. Feature weights are calculated by average of 100,000 random validation example. The weights in bold blue are for the best selected features according to Sharafaldin et al. [20].



(a) Without multi-hinge loss. (b) With multi-hinge loss.

Fig. 5: Visualization of data before softmax layer.

Experiment of Multi-Hinge Loss: In this experiment, to analyze the effect of hinge loss in model training, we visualized 10,000 random inputs in validation set represented by two models.Input examples are fed-forward into the deep learning model, extracted the final hidden layer representation, transformed into lower-dimensional space via t-SNE [17] and plotted in Figure 5. In Figure 5a, data is represented by the without-hinge-loss model, all data points distributed into a sphere. The data has a certain cluster resolution, but there is large interference between clusters. In Figure 5b, data is represented by the with-hinge-loss model, the data representation space has doubled from $[-2, 2]$ to $[-4, 4]$. We have observed that data belonging to the same class has been clustered closer together and these clusters also tend to be further apart. This is consistent with the idea of hinge loss, which is to maximize the margin between the data classes. As shown in Figure 2, the hinge loss-trained model is less likely to misclassify classes when compared to the model that trained on cross entropy loss only.

4 Related Work

Kaur et al. [13] classifies DDOS detection methods into two main groups: signature-based detection and anomaly-based detection. Detection with signature-based makes utilization of 'signs' about different attacks. This approach is only operative in case of known attack; it works by matching the arriving traffic with the previously-stored pattern.Anomaly-based detection methods are more commonly used since it is fit for recognizing unknown attack. The main strategy is comparing standard network performance with arriving information to detect anomalies, i.e., when a system is under DDoS attack, unexpected fluctuations in the network traffic need to be noticed. Since DDoS attacks are still growing year by year, knowledge-based methods are inflexible to adapt to their growth and change. The research community has been paying attention to DDoS detection for years, provided several different methods for recognizing DDoS attacks based on statistical and Machine Learning techniques.

Statistical methods are basically done by measuring statistical properties (i.e., means and variances) of various normal traffic parameters. Three of the most widely used techniques in these approaches are ARIMA [23], SSM [18] and CAT-DCP [2]. The limitation of these methods is they are not able to determine with certainty the normal network packet distribution.

Machine learning-based techniques are useful as they do not have any prior known data distribution. The machine learning methods used for DDoS detection are very diverse: Support Vector Machine [20, 12], Naive Bayes [6], Decision Tree [22]. Tradition machine learning methods require selecting the best feature-set to bring good performance i.e., they often require the contribution of human experts high level to define patterns. This process is labor-intensive, comparatively expensive but often provide much error-prone [21].

Neural networks are introduced as an alternative to traditional machine learning methods that can handle complex data by automatically select useful features without human intervention. Many deep neural network-based methods

have been successfully applied in many works for generating intrusion detection classifiers in general and DDOS detection in particular. Examples include Artificial Neural Network [19], Convolutional Neural Network, Recurrent Neural Network (RNN) and its improvements Long Short-Term Memory Unit (LSTM) and Gated Recurrent Units (GRUs) [1], Hopfield Networks and Radial Basis Function based Neural Networks [11], Replicator Neural Networks [18], Convolutional Neural Network [16], etc.

Although the deep learning-based model has recently achieved great success due to its high learning capacity, it still cannot escape from imbalanced data [9, 10]. To overcome this problem, two methods that have been successfully applied in other domains are using hinge loss [9] and applying class-weight to give priority to the minor classes [10].

5 Conclusion

We have proposed a DDoS attack detection and classification model that takes advantage of advanced deep learning techniques. It starts with an automatic feature-selection component based on the context of the input feature set. The weighted features are classified with a fully-connected MLP with softmax activation. Our models has been trained with objective function is combination of cross entropy and hinge loss.

The experiments on CIC-DDoS 2019 datasets has demonstrated the effectiveness of our proposed model when compared with other comparative machine learning-based and neural neural network-based models. We also investigated and verified the rationality and contributions of automatic feature selection models. Results have also shown the effectiveness of weighted loss and hinge loss in dealing with class imbalance problems.

Our limitation rare and hierarchical labels is highlighted since it resulted in low performance on UDPLag class. Furthermore, the discrepancy between the data distribution in training and testing data also led to poor results for some labels. We aim to address these problems in our future works.

References

1. Aldweesh, A., Derhab, A., Emam, A.Z.: Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. Knowledge-Based Systems **189**, 105124 (2020)
2. Chen, Y., Hwang, K., Ku, W.S.: Collaborative detection of ddos attacks over multiple network domains. IEEE Transactions on Parallel and Distributed Systems **18**(12), 1649–1662 (2007)
3. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. Journal of machine learning research **2**(Dec), 265–292 (2001)
4. Douligeris, C., Mitrokotsa, A.: Ddos attacks and defense mechanisms: classification and state-of-the-art. Computer Networks **44**(5), 643–666 (2004)

5. Elsayed, M.S., Le-Khac, N.A., Dev, S., Jurcut, A.D.: Ddosnet: A deep-learning model for detecting network attacks. In: WoWMoM. pp. 391–396. IEEE (2020)
6. Fouladi, R.F., Kayatas, C.E., Anarim, E.: Frequency based ddos attack detection approach using naive bayes classification. In: 2016 39th International Conference on Telecommunications and Signal Processing (TSP). pp. 104–107. IEEE (2016)
7. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 249–256 (2010)
8. Gyanchandani, M., Rana, J., Yadav, R.: Taxonomy of anomaly based intrusion detection system: a review. International Journal of Scientific and Research Publications **2**(12), 1–13 (2012)
9. Huang, C., Li, Y., Loy, C.C., Tang, X.: Learning deep representation for imbalanced classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5375–5384 (2016)
10. Johnson, J.M., Khoshgoftaar, T.M.: Survey on deep learning with class imbalance. Journal of Big Data **6**(1), 27 (2019)
11. Karimazad, R., Faraahi, A.: An anomaly-based method for ddos attacks detection using rbf neural networks. In: Proceedings of the International Conference on Network and Electronics Engineering. vol. 11 (2011)
12. Kato, K., Klyuev, V.: An intelligent ddos attack detection system using packet analysis and support vector machine. IJICR **14**(5), 3 (2014)
13. Kaur, P., Kumar, M., Bhandari, A.: A review of detection approaches for distributed denial of service attacks. Systems Science & Control Engineering **5**(1), 301–320 (2017)
14. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
15. Li, Q., Meng, L., Zhang, Y., Yan, J.: Ddos attacks detection using machine learning algorithms. In: International Forum on Digital TV and Wireless Multimedia Communications. pp. 205–216. Springer (2018)
16. Liu, Y.: Ddos attack detection via multi-scale convolutional neural network. Computers, Materials & Continua **62**(3), 1317–1333 (2020)
17. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
18. Prasad, K.M., Reddy, A.R.M., Rao, K.V.: Dos and ddos attacks: defense, detection and traceback mechanisms-a survey. Global Journal of Computer Science and Technology (2014)
19. Saied, A., Overill, R.E., Radzik, T.: Detection of known and unknown ddos attacks using artificial neural networks. Neurocomputing **172**, 385–393 (2016)
20. Sharafaldin, I., Lashkari, A.H., Hakak, S., Ghorbani, A.A.: Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. In: 2019 ICCST. pp. 1–8. IEEE (2019)
21. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE transactions on emerging topics in computational intelligence **2**(1), 41–50 (2018)
22. Wu, Y.C., Tseng, H.R., Yang, W., Jan, R.H.: Ddos detection and traceback with decision tree and grey relational analysis. International Journal of Ad Hoc and Ubiquitous Computing **7**(2), 121–136 (2011)
23. Zhang, G., Jiang, S., Wei, G., Guan, Q.: A prediction-based detection algorithm against distributed denial-of-service attacks. In: Proceedings of the 2009 international conference on wireless communications and mobile computing: Connecting the World wirelessly. pp. 106–110 (2009)