The 35th Canadian Conference on Artificial Intelligence
DOI: 0

Canadian AI 2022
30 May–3 June 2022, Toronto

DTEXNet: Artificial Intelligence-Based Combination Scheme for DDoS Attacks Detection

Salma Elgendy[†], Mayar Attawiya[†], Omar Haridy[†,*], Ahmed Farag[†], Paula Branco[†]
[†] EECS - Faculty of Engineering, University of Ottawa, Canada

Abstract

Distributed Denial of Service (DDoS) attacks became the most widely spread attack because of their ease of design and execution. Such attacks are challenging to detect and mitigate due to the diversity of DDoS attack modes and the variable size of attack traffic. This makes the research on DDoS attack detection extremely important. Machine learning techniques are used to detect various DDoS attacks with complex and dynamic patterns. However, such techniques require extensive pre-processing and feature engineering to the data to achieve acceptable results. On the other hand, neural networks can achieve acceptable results without the need for such prior preparations. This paper proposes a novel combination scheme between EfficientNet, Xception, and Decision Tree models called DTEXNet. DTEXNet combines two neural networks to benefit from their ability to extract features without the need to prior preparations, and a classical machine learning model that has high performance on similar problems. The solution proposed uses two convolutional neural networks (CNNs) to classify between 10 types of DDoS attacks and uses their prediction results to enhance the performance of Decision Tree model on the same classification task. The results of the experiments carried out show that the proposed solution can significantly improve the results of the Decision tree, EfficientNet, and Xception models if applied individually.

Keywords: Distributed Denial of Service, Deep Learning, Convolutional Neural Networks, Decision Tree, EfficientNet, Xception

1. Introduction

Distributed Denial of Service (DDoS) attacks are one of the major cyber threats on communications networks and have dramatically increased over the past years. The high frequency of DDoS attacks is due to their simplicity while being cheap to initiate. However, the impact of these attacks on the victims can be potentially severe and produce serious losses. These attacks overwhelm the victims and making them unable to provide their services normally [1]. One of the examples of DDoS attacks is in the field of autonomous and connected vehicles that involves sharing information between vehicles and infrastructures through wireless communication. The risk of an attack increases as vehicles become more connected through wireless networks that allow vehicle-to-infrastructure (V2I) communication. Attacks on V2I communication can have serious consequences if the systems are not properly secured. Still, V2I applications have multiple vulnerabilities that attract hackers [2]. One example of V2I communication is autonomous parking system in smart garages, which relies on vehicle to garage communication (V2G), putting the whole system under the risk of potential DDoS attacks through Wi-Fi. Attacks on autonomous vehicles are very dangerous as the car might become blind in the presence of pedestrians or other vehicles, leading to huge damages. Thus, the timely detection of these attacks is one of the main defense mechanisms. It is known that traditional security solutions such as firewalls and intrusion detection systems are unable to detect the complex DoS/DDoS attacks; most of these solutions filter the normal and suspicious traffic using only predefined rules [3].

[*]omar.abdelbaset2057@gmail.com

The use of artificial intelligence (AI)-based solutions has enabled researchers to detect DDoS attacks with complex and dynamic patterns. In this context, multiple classical machine learning methods such as Naïve Bayes, Decision trees, and K-Nearest Neighbors were used to detect DDoS and DoS attacks. For instance, Decision Trees (DT) were recently used to detect these attacks, and can either be used on their own or in combination with other models (e.g. [4, 5]). However, classical machine learning models need extensive preprocessing and feature engineering and selection. On the other hand, neural networks do not require such feature engineering since they can achieve good performance in classification by directly feeding the data into the network.

In this paper, we propose a novel AI-based algorithm named DTEXNet (**D**ecision **T**ree with **E**fficientNet and **X**ception **Net**work features). DTEXNet consists of three key stages. The first stage involves the generation of images representation for the dataset cases. This stage will allow the use of CNNs in the following stages. In Stage 2 the prediction scores of two neural network models are obtained and combined with the original feature set to form a new set of informative features. The final third stage includes using a classical machine learning model to classify the attacks based on the newly created feature set. DTEXNet builds on top of current detection models available making a combination between neural networks which have high performance, and classical machine learning models that can be tested to find the best models meeting the problem deployment requirements. This results in a new detection model that achieve an effective detection of DDoS attacks. For the neural networks, state-of-the-art models ([3] and [6]) are used to build a model using convolutional neural networks (CNN). For the machine learning part, a tuned Decision Tree is used as it has shown great performance in solving similar problems. We selected two recently published datasets to carry out our experiments: the DDoS Evaluation Dataset (CIC-DDoS2019)[1] and the Intrusion Detection Evaluation Dataset (CIC-IDS2017)[2].

Our main contributions are as follows:

- provide a literature review on DDoS attack detection using artificial intelligence-based solutions;
- present a novel solution DTEXNet that combines two CNNs with slight modifications and a Decision Tree model to detect DDoS attacks; and
- carry out an extensive performance evaluation that shows the advantages of DTEXNet against a classical Decision Tree and two CNN algorithms in different DDoS attacks datasets.

This paper is organized as follows. In Section 2 we provide a literature review on solving similar classification problems using different neural networks-based and classical machine learning models-based solutions. Section 3 presents DTEXNet, our novel proposed solution for DDoS detection that is based on a combination scheme between CNNs and Decision Tree. In Section 4 we describe the experiments that were carried out and discuss the results obtained. Finally, Section 5 concludes this paper and provides future research directions.

2. Related Work

Machine learning is used widely nowadays for detecting attacks enabling researchers to detect DDoS attacks with complex and dynamic patterns. Traditional solutions alone like firewalls are no longer able to detect complex DoS and DDoS attacks. Neural networks (NN) and biological danger theory were used to reduce DDoS attacks in the context of Software Defined Network (SDN) [7]. Radial basis function support vector machine (RBF-SVM) and linear classification SVM for DDoS detection were also experimented ([8, 9]);

[1]https://www.unb.ca/cic/datasets/ddos-2019.html
[2]https://www.unb.ca/cic/datasets/ids-2017.html

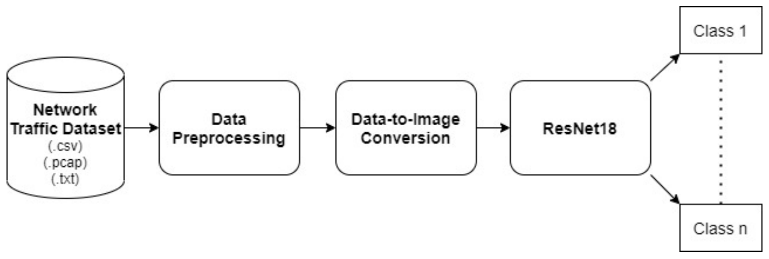

Figure 1. Methodology used in [3]

Authors	Reference	Precision	Recall	F1-Score
Sharafaldin, Lashkari, Hakak, and Ghorbani	[13]	78%	65%	69%
Hussain, Abbas, Husnain, Fayyaz, Shahzad, and Shah	[3]	87%	86%	86%

Table 1. Comparison between the solutions presented in [3] and [13].

the accuracy rates in these studies reached 95.11% and 97.60%, respectively. In [10] an accuracy of 97.6% with a false positive rate of 3.85% was reached using multiple linear SVM and Self Organizing Map (SOM). Several other standard machine learning algorithms such as decision trees, support vector machines, or gradient boosting have been used (e.g. [11, 12]). However, the use of these strategies always requires complex feature engineering and extensive data preparation which prevents their use in practice.

Neural networks, on the other hand, does not require feature engineering achieving a good performance by directly inputting the data into the network. Moreover, neural networks methods have shown high efficiency in multiple domains. The first paper discussed for their state-of-the-art technique in neural networks uses ResNet in IoT DoS and DDoS detection [3]. It is also crucial to mention that they are working on the CICDDoS2019 dataset, which is one of our selected datasets. During the last years, deep learning, especially convolutional neural networks (CNN), made high accomplishments in the image processing field so the authors concluded that the potential of using CNNs in DDoS attacks detection will be efficient by converting the network traffic data into three-channel image form. They proposed a methodology to convert the network traffic data into image form and trained a state-of-the-art CNN model, i.e., ResNet over the converted data.

Data Preprocessing is crucial to clean the tabular data. Data preprocessing includes dropping the records with missing data and dropping duplicated feature columns. After the data preprocessing step, the dataset ended up with 60 features and the samples were ready to be converted into image format in the data conversion process. For the Data Conversion, each feature is normalized, and the dataset is separated into two data frames, attack and normal. Data frames are divided into chunks of 180 records, each record consists of 60 pre-processed features and are finally converted into a 60x60x3 image shape to be finally fed to the ResNet18 for classification. Using this methodology, they reached 99.9% accuracy in binary classification, and 87% average precision recognizing eleven types of DoS and DDoS attack patterns which is 9% higher as compared to the state-of-the-art as shown in Table 1.

For the latest paper using CICDDoS2019 dataset, Wang H et al. designed a hybrid neural network DDosTC structure, combining efficient and scalable transformers and a CNN to detect DDoS attacks on SDN, tested on the CICDDoS2019 data [6]. They conducted several experiments on different state-of-the-art deep learning models in the field of DDoS intrusion detection. The experimental results show that the average AUC of DDosTC is 2.52% higher than the current optimal model and that DDosTC is more successful than the current optimal model in terms of average accuracy, average recall, and F1 score. Their solution was to use deep learning transformers plus CNNs DDoS attack-detection model "DDosTC". The detection module is implemented on the control plane, as it is considered the brain of the SDN. The transformer is composed of an encoder and decoder. Based on the accurate results obtained by using Machine Learning models that need extensive feature engineering, such as RBF-SVM in [8] and [9], DT in [4] and [5], and the novel CNN-based method in [3] and [6] that converts the network traffic tabular data into images, the proposed solution is designed to form a combination scheme between both DT and CNNs to eliminate the need for feature engineering, in addition to enhancing the state-of-the-art accuracy obtained by the DTs for solving DDoS attacks Detection problem. DETEXNet uses several CNNs to get the probability of occurrence of each attack, then uses the output of each CNN to form feature vectors that can be concatenated as feature columns to the original tabular network traffic data, and then feeds the modified dataset to the DT.

3. The DTEXNet Algorithm

Our proposed solution, DTEXNet, is a combination of neural networks and a classical machine learning model that allows to benefit from the advantages of each. First, a CNN is used. Since it is a deep learning model it does not require feature extraction to be carried out. CNN is specifically chosen to take advantage of its great performance on image datasets by converting the chosen dataset to 3-channel images [3]. The solution consists of three main stages. Stage 1 includes converting the traffic data to images. In Stage 2 the two selected CNNs, modified version of EfficientNet and Xception, are used to obtain a new feature set that is concatenated with the original feature set; hence, a new modified dataset is formed that will then be fed into Stage 3. A Decision Tree is trained on the newly created feature set and the predictions of the final classification model is obtained on the test set. Algorithm 1 shows the pseudo code for the DTEXNet Algorithm displaying its main stages.

Algorithm 1 The DTEXNet Algorithm.

```
1: Normalize data using min-max normalization                              ▷ Initialize Stage 1
2: Convert data into images
3: Train modified EfficientNet                                             ▷ Initialize Stage 2
4: obtain EfficientNet prediction probabilities
5: Train the modified Xception Network
6: obtain the Xception prediction probabilities
7: Concatenate original dataset features with features from EffcientNet and Xception
8: Train a Decision Tree with the new feature set                          ▷ Initialize Stage 3
9: obtain the predictions for the test set using the DT model
```

3.1. DTEXNet Stage 1: Data Preprocessing

Data must be prepared for the CNN models. This includes cleaning the dataset and converting it into images. Regarding the cleaning process we removed constant features from the datasets and removed the rows with missing values or noisy information such as "Not A Number" (NAN) and infinity (inf) values.

CNNs have shown great performance on images, so it will be an added value to use CNNs in intrusion detection modules. However, in order to use CNNs it is necessary to transform their inputs into images. Since the data for intrusion detection is usually provided in *csv* format, it has to be converted to *RGB* images to be used with CNNs. Figure 2 shows the pipeline used to convert the samples in *csv* format to images. First, the dataset is sorted by labels so that the data is in sequence, where the samples of each class are after one

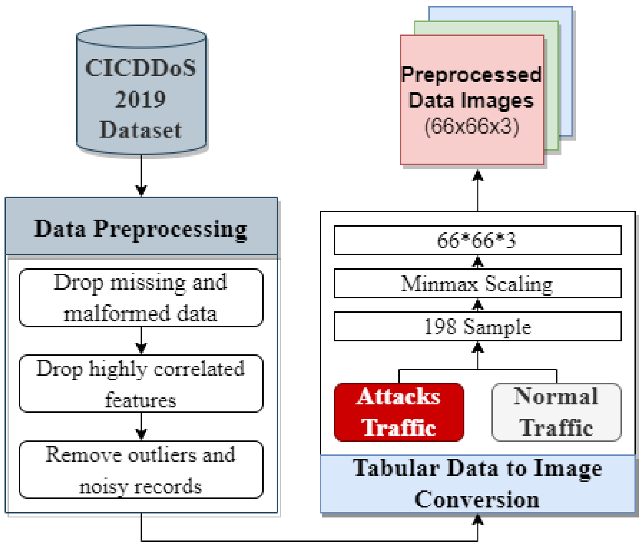
CICDDoS 2019 Dataset
Data Preprocessing
Drop missing and malformed data
Drop highly correlated features
Remove outliers and noisy records
Preprocessed Data Images (66x66x3)
66*66*3
Minmax Scaling
198 Sample
Attacks Traffic
Normal Traffic
Tabular Data to Image Conversion

Figure 2. Example of images preparation pipeline for Stage 1 on CIC-DDoS2019 data.

another. Then, the feature columns are normalized using min-max normalization as shown in the following equation: $x' = \frac{x-min(x)}{max(x)-min(x)} \times 255$. Then, the samples are arranged in an image format. For a dataset with a given number of features F, images are created with size $F \times F \times 3$, since the selected CNN models require RGB images. This means that, the number of samples required to form a single image equals to $F \times 3$. So, the data must be divided into chunks of $F \times 3$. The samples required to form an image are taken from the same class so that the image created belongs to that class. To achieve this, the samples are first sorted by label. For example, in CIC-DDoS2019, the dataset has 66 features, hence, 66 samples are required to form one image channel of size 66×66. Moreover, two more channels need to be created for RGB, which means 198 samples from the same class in total will be needed to form an image of size $66 \times 66 \times 3$. Figure 2 displays these calculated dimensions for the CIC-DDOS2019 dataset. From the sorted dataset, a chunk of 198 samples is taken from the same class to form each image. Figures 3 and 4 show samples of the generated images from the CIC-DDoS2019 dataset.

3.2. **DTEXNet Stage 2: New feature set through CNNs**

CNNs were selected with the goal of achieving the highest accuracy on the required classification task. The CNN models selected were chosen for their capability of extracting features efficiently, each by its own unique method, with suitable number of parameters that allows the CNN to be efficiently trained using the available hardware. The first CNN model is EfficientNet-B0, a mobile-sized CNN model. EfficientNet-B0 was built with a goal to optimize both accuracy and floating point operations (FLOPS) represented as: $ACC(m) * (FLOPS(m)/T)^{\omega}$, where $ACC(m)$ and $FLOPS(m)$ denote the accuracy and FLOPS of model, T is the target FLOPS, and $\omega = -0.07$ is a hyperparameter responsible for the trade-off between accuracy and FLOPS [14].

EfficientNet-B0 requires input images of size 224×224×3. As a result, the output images from the data conversion module need to be resized to 224×224×3. EfficientNet-B0 was selected as it requires the smallest input size and has the least number of parameters (5,330,571 parameters). The convolutional layers' structure of EfficientNet-B0 is used unchanged. However, several layers were added to make the model fit the requirements of

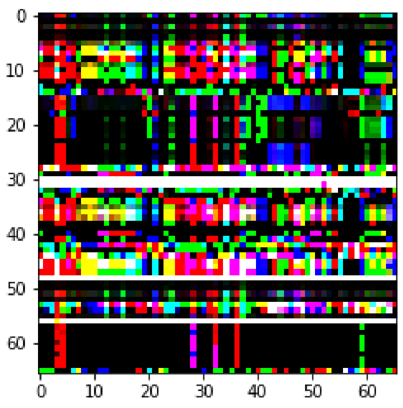

Figure 3. Image created from class BENIGN of CIC-DDoS2019.

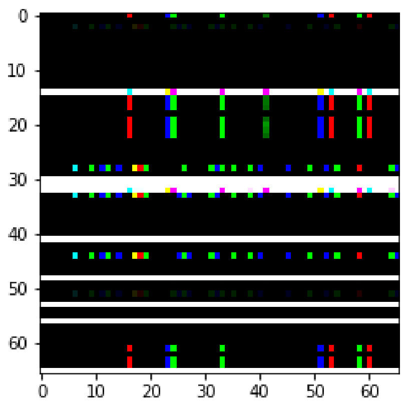

Figure 4. Image created from class DDoS_MSSQL class of CIC-DDoS2019.

Stage i	Added Layers to EfficientNet-B0 Architecture	
	Operator $\hat{F}_i$	Corresponding Parameter
1	Dropout regularization	Rate = 0.2
2	Fully Connected Layer (Dense)	Output neurons = 11, Activation = Softmax
3	L2 regularization	Rate = 0.0001

Table 2. Added layers to EfficientNet-B0 architecture

the current classification problem. Additional layers were added as shown in Table 2. All parameters of EfficientNet-B0 are retrained on the dataset using stochastic gradient descent (SGD) optimizer and categorical cross entropy loss function.

The second CNN model used is Xception [15], a CNN model inspired by Inception modules. Inception modules are used in CNN to allow for more efficient computation and deeper networks through dimensionality reduction technique using stacked 1×1 convolutions, which makes the CNN capable of learning richer representations with less parameters. Xception is a CNN based entirely on depthwise separable convolution layers and it makes the hypothesis that the mapping of cross-channels correlations and spatial correlations in the feature map of convolutional neural networks can be entirely decoupled. This CNN architecture contains 22,910,480 parameters. The convolutional layers' structure of Xception is used unchanged. However, several layers were added to make the model fit the requirements of the current classification problem. The additional layers are shown in Table 3.

All parameters of Xception are retrained using Adam optimizer and categorical cross entropy loss function. Stage 2 ends with the concatenation of the features obtained through the two modified networks and the original feature set. This newly developed set of features will be used in Stage 3 to learn the final predictive model.

3.3. **DTEXNet Stage 3: DT training**

We selected a Decision Tree as the final model to be trained. DT is one of the quickest ways to identify relationships between variables and the most significant variable. It is also an interpretable model that may be used to provide insights regarding the important features being used to solve the predictive task. The parameters used for the DT model are listed in

Operator	Corresponding Parameter
Fully Connected Layer (Dense)	Output neurons =1024 Activation= Relu
Batch normalization	-
Dropout regularization	Rate = 0.3
Fully Connected Layer (Dense)	Output neurons =1024 Activation= Relu
Batch normalization	-
Dropout regularization	Rate = 0.3
Fully Connected Layer (Dense)	Output neurons =1024 Activation= Relu
Fully Connected Layer (Dense)	Output neurons =11 Activation= Softmax

Table 3. Added layers to Xception architecture.

Parameter	Value
n_estimators	100
criterion	"entropy"
max_depth, n_jobs	
random_state, class_weight	None
max_samples	
max_leaf_nodes	200
min_samples_split	2
min_samples_leaf	1
min_weight_fraction_leaf	
ccp_alpha	0.0
min_impurity_decrease	
max_features	"auto"
bootstrap	True
oob_score, warm_start	False

Table 4. DT parameters.

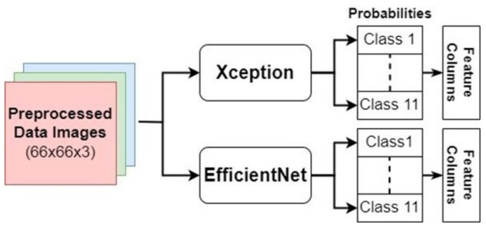

Figure 5. Stage 1 of the classification pipeline (CNN training)

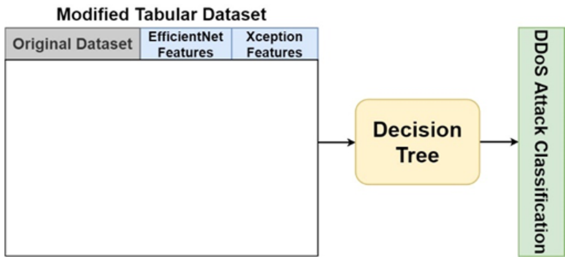

Figure 6. Stage 2 of classification pipeline: original data and 2 CNNs features concatenated and fed to DT

Table 4. The DT model is then trained on the newly built dataset with the concatenated features. Figures 5 and 6 provide an overview of the proposed DTEXNet Algorithm when applied on CIC-DDoS2019 dataset.

As illustrated in Figure 5, a modified EfficientNet-B0 is trained with the converted images, then the probabilities produced from the prediction of the training and testing sets are obtained. As explained in Stage 1, each image consists of $F \times 3$ samples. As a result, each probability vector obtained by the CNNs is repeated $F \times 3$ times. For example, in the CIC-DDoS2019 dataset, each vector of the sorted probabilities is repeated 198 times, because each image consists of 198 samples as discussed earlier. Finally, these probabilities are concatenated to the original preprocessed training and testing feature sets. The same procedures are repeated with the modified Xception Network. The last step is to train the decision tree with the newly created dataset. This model will take advantage of the newly extracted features to predict the attacks on the test set.

4. **Experiments and Results**

4.1. **Datasets**

To evaluate the effectiveness of the DTEXNet Algorithm, we selected two datasets: the DDoS Evaluation Dataset (CIC-DDoS2019) and the Intrusion Detection Evaluation Dataset (CIC-IDS2017) from the Canadian Institute for Cybersecurity. These datasets consist of real-time network traffic, are new, extensive and versatile, and contain both inbound and outbound traffic of the latest intrusion attacks. Both are multiclass datasets containing several attacks and a benign class.

The CIC-DDoS2019 dataset [13] contains six different types of DDoS attacks and others resulting in 11 different attack types, additionally normal traffic, so 12 classes in total. It originally had 88 features, but after removing features with low correlation with the label column or constant value features, 66 unique and important features remained.A subset of 100,000 samples was used from each attack type of this dataset to deal with the large size of the data. All the samples of the benign class are used, and the "TFTP" data, which was 9 GB alone, is discarded. As a result, the subset used in this paper has 11 classes in total.

The second dataset is the CIC-IDS2017 dataset [16], a reliable benchmark dataset. This dataset contains 2830540 samples of normal and attack traffic data spanned over five days.The number of features of this dataset are originally 83 features, but 18 highly correlated features were dropped. The number of attack classes were originally 14, but 5 attack classes were dropped for their low number of records. The dropped classes are Heartbleed (11 samples), three types of web attacks (2180 samples), and Infiltration attacks (36 samples). Hence, the number of classes of the filtered dataset are nine attack classes and one benign class. The CIC-IDS2017 is a highly imbalanced dataset, unlike the CIC-DDoS2019. This required an extra procedure in the preprocessing pipeline which is the use of Borderline SMOTE oversampling technique. This method was applied on the training data to solve the imbalance problem.

The number of samples remained after the preprocessing is 875,300 and 108,184 samples in the training and testing subsets of the CIC-DDoS2019 dataset respectively, while 696,149 and 128,115 samples remained in the training and testing subsets of the CIC-IDS2017 dataset.

4.2. **Experimental Settings**

The holdout method was used to estimate the models' performance, where the each dataset is split into 80% training data and 20% testing subsets. We opted for this strategy due to the relatively large size of the datasets. This study's experiments include the evaluation of the following models: tuned decision tree results, Xception and EfficientNet-B0 results, and our proposed DTEXNet results.

All experiments took place on Tesla P100 GPUs with 25 GB RAM. The metrics used to evaluate the models' performance are the precision, recall, and F1 scores. Given that the classification problem is a multi-class task the metrics are calculated for each class. We selected these metrics as they are useful when evaluating the performance under imbalanced domains [17, 18].

4.3. **Results and Discussion**

A baseline decision tree is trained and tested on both datasets. Table 5 displays the DT precision, recall, F1 scores applied on the testing subset of CIC-DDoS2019 dataset. Table 6 displays the DT results for the CIC-IDS2017 dataset.

For testing the performance of EfficientNet-B0 and Xception Network, the datasets were transformed into images. In CICDDoS2019, there are 875,300 training samples which will be divided into chunks of 198 samples, hence, a total of 4,420 images of size 66x66x3 will be created. Similarly, 546 testing images will be created. In CIC-IDS2017, a chunk of 195 samples creates an image resulting in 3,570 training images and 657 testing images, each of size 65x65x3.

Figures 7, 8, 9 and 10 show the learning curves of Xception and EfficientNet using CIC-DDoS2019 dataset. It can be observed that as the model trains on more epochs, its training and validation accuracies and losses become closer in value.

Label: Class	Precision	Recall	F1-score
0: BENIGN	0.99	0.99	1.00
1: DrDoS_DNS	0.91	0.88	0.90
2: DrDoS_LDAP	0.82	0.74	0.79
3: DrDoS_MSSQL	0.96	0.95	0.96
4: DrDoS_NTP	0.99	1.00	1.00
5: DrDoS_NetBIOS	1.00	1.00	1.00
6: DrDoS_SNMP	0.8	0.91	0.86
7: DrDoS_SSDP	0.64	0.56	0.60
8: DrDoS_UDP	0.61	0.69	0.65
9: Syn	0.52	0.94	0.66
10: UDP-lag	0.67	0.13	0.21
Macro Average	0.81	0.80	0.78

Table 5. Decision Tree scores on CIC-DDoS2019 dataset.

Label: Class	Precision	Recall	F1-score
0: BENIGN	0.97	0.95	0.96
1: Bot	0.45	1.00	0.62
2: DDoS	1.00	1.00	1.00
3: DoS GoldenEye	0.89	0.97	0.93
4: DoS Hulk	1.00	0.98	0.99
5: DoS Slowht	0.65	0.86	0.74
6: DoS Slowloris	0.87	0.70	0.78
7: FTP-Patator	0.98	1.00	0.99
8: PortScan	1.00	0.99	1.00
9: SSH-Patator	0.81	1.00	0.90
Macro Average	0.86	0.94	0.89

Table 6. Decision Tree scores on CIC-IDS2017 dataset.

Label: Class	Precision	Recall	F1-score
0: BENIGN	1.00	1.00	1.00
1: DrDoS_DNS	0.98	1.00	0.99
2: DrDoS_LDAP	1.00	0.98	0.99
3: DrDoS_MSSQL	0.93	1.00	0.96
4: DrDoS_NTP	1.00	0.88	0.93
5: DrDoS_NetBIOS	0.98	1.00	0.99
6: DrDoS_SNMP	0.98	0.98	0.98
7: DrDoS_SSDP	0.78	0.90	0.85
8: DrDoS_UDP	0.83	0.61	0.71
9: Syn	0.90	0.88	0.89
10: UDP-lag	0.92	0.93	0.92
Macro Average	0.93	0.92	0.92

Table 7. EfficientNet-B0 scores on CIC-DDoS2019 dataset.

Label: Class	Precision	Recall	F1-score
0: BENIGN	1.00	0.83	0.91
1: Bot	1.00	1.00	1.00
2: DDoS	0.98	1.00	0.99
3: DoS GoldenEye	0.88	0.93	0.90
4: DoS Hulk	0.99	1.00	1.00
5: DoS Slowht	0.19	0.62	0.29
6: DoS slowloris	0.89	1.00	0.94
7: FTP-Patator	1.00	1.00	1.00
8: PortScan	1.00	1.00	1.00
9: SSH-Patator	1.00	1.00	1.00
Macro Average	0.89	0.93	0.80

Table 8. EfficientNet-B0 scores on CIC-IDS2017 dataset.

EfficientNet-B0 is trained and tested on the converted CIC-DDoS2019 dataset. Table 7 displays the precision, recall, and F1 scores of each class. Table 8 displays the EfficientNet-B0 precision, recall, and F1 scores applied on CIC-IDS2017 dataset.

The second network Xception was also tested under the same conditions. Xception was trained and tested on the converted CIC-DDoS2019 dataset. Tables 9 and 10 display the precision, recall, and F1 scores of each class on CIC-DDoS2019 and CIC-IDS2017 datasets..

Finally, our DTEXNet solution was evaluated on both datasets. The probabilities estimated by EfficientNet-B0 and Xception were concatenated with the original cleaned data as additional features, as previously mentioned. This resulted in a training set of shape (875160, 88), and a testing set of shape (108108, 88) for the CIC-DDoS2019 dataset. In CIC-IDS2017 dataset, the training set had a shape (696149, 85) and the testing set's shape was (128115, 85). A Decision tree is then trained and tested on the newly generated dataset. Tables 11 and 12 display the DTEXNet precision, recall, and F1 scores of each class applied to CIC-DDoS2019 and CIC-IDS2017 datasets, respectively.

The results of the baseline DT model shows that the model did not overfit the data but still needs considerable improvement to efficiently detect the attacks. This performance may be the result of the inability of the decision tree to distinguish between similar attacks, or to identify classes with complex features, resulting in low F1 scores to some classes. DT classified 6 classes with F1 scores below 90% where the lowest F1 score was 21% in CIC-DDoS2019 dataset. Moreover, DT classified 4 classes with F1 scores below 90% where the

Label: Class	Precision	Recall	F1-score
0: BENIGN	1.00	0.97	0.99
1: DrDoS_DNS	0.99	0.98	0.99
2: DrDoS_LDAP	0.99	1.00	1.00
3: DrDoS_MSSQL	1.00	0.99	0.99
4: DrDoS_NTP	1.00	1.00	1.00
5: DrDoS_NetBIOS	1.00	0.99	0.99
6: DrDoS_SNMP	0.99	1.00	0.99
7: DrDoS_SSDP	0.55	0.80	0.65
8: DrDoS_UDP	0.76	0.51	0.61
9: Syn	0.92	0.88	0.90
10: UDP-lag	0.88	0.92	0.90
Macro Average	0.92	0.91	0.91

Table 9. Xception scores on CIC-DDoS2019 dataset.

Label: Class	Precision	Recall	F1-score
0: BENIGN	0.97	1.00	0.99
1: Bot	0.38	1.00	0.55
2: DDoS	1.00	1.00	1.00
3: DoS GoldenEye	1.00	1.00	1.00
4: DoS Hulk	1.00	1.00	1.00
5: DoS Slowht	1.00	1.00	1.00
6: DoS Slowloris	1.00	1.00	1.00
7: FTP-Patator	1.00	1.00	1.00
8: PortScan	1.00	0.99	1.00
9: SSH-Patator	0.00	0.00	0.00
Macro Average	0.83	0.90	0.85

Table 10. Xception scores on CIC-IDS2017 dataset.

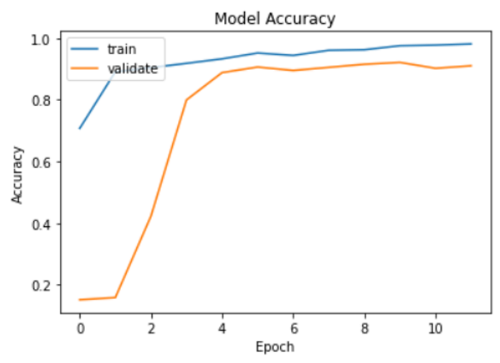

Figure 7. Xception accuracy learning curve

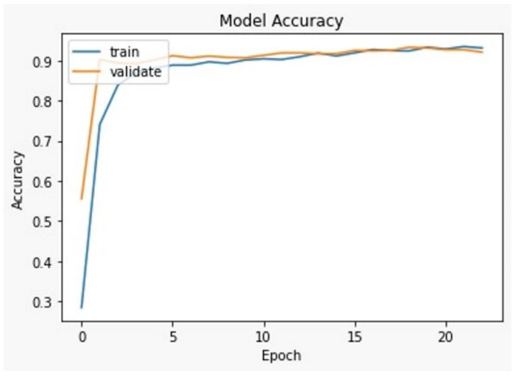

Figure 8. EfficientNet accuracy learning curve

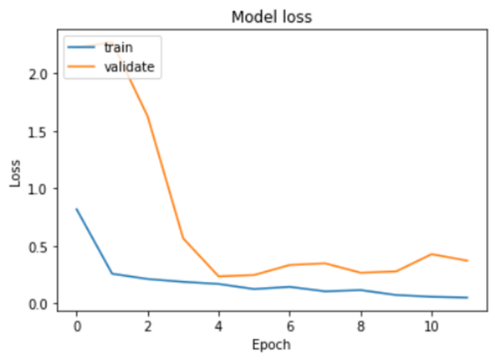

Figure 9. Xceptiom loss learning curve

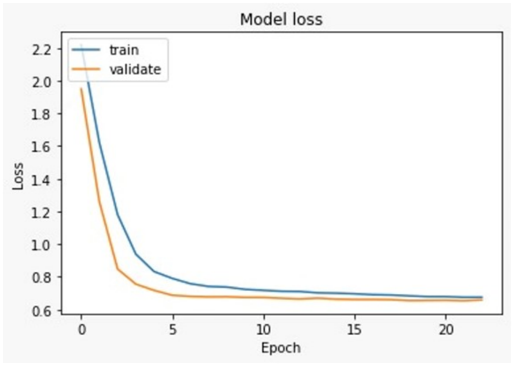

Figure 10. EfficientNet-B0 loss learning curve

lowest F1 score was 62% in CIC-IDS2017 dataset. On the other hand, CNNs showed better results in classifying the attacks. However, EfficientNet-B0 had difficulties in distinguishing some classes, where it scored F1-scores below 90% in 2 classes in CIC-DDoS2019 dataset and the lowest F1 score was equal to 80%. Moreover, EfficientNet-B0 had difficulties in classifying 2 classes in CIC-IDS2017 dataset with the lowest F1 score equal to 29%. Xception

Label: Class	Precision	Recall	F1-score
0: BENIGN	1.00	1.00	1.00
1: DrDoS_DNS	0.99	1.00	1.00
2: DrDoS_LDAP	1.00	0.98	0.99
3: DrDoS_MSSQL	1.00	0.99	0.99
4: DrDoS_NTP	0.96	1.00	0.98
5: DrDoS_NetBIOS	0.99	1.00	1.00
6: DrDoS_SNMP	0.98	0.97	0.98
7: DrDoS_SSDP	0.75	0.73	0.74
8: DrDoS_UDP	0.74	0.74	0.74
9: Syn	1.00	1.00	1.00
10: UDP-lag	0.99	1.00	0.99
Macro Average	0.95	0.95	0.95

Table 11. DTEXNet scores on CIC-DDoS2019 dataset.

Label: Class	Precision	Recall	F1-score
0: BENIGN	1.00	1.00	1.00
1: Bot	1.00	1.00	1.00
2: DDoS	1.00	1.00	1.00
3: DoS GoldenEye	1.00	0.60	0.75
4: DoS Hulk	1.00	1.00	1.00
5: DoS Slowht	1.00	1.00	1.00
6: DoS Slowloris	1.00	1.00	1.00
7: FTP-Patator	0.61	1.00	0.76
8: PortScan	1.00	0.99	1.00
9: SSH-Patator	1.00	1.00	1.00
Macro Average	0.96	0.96	0.95

Table 12. DTEXNet scores on CIC-IDS2017 dataset.

also had some difficulties in the classification of 2 classes of CIC-DDoS2019 dataset, where it exhibited F1-scores below 90% and the lowest F1 score was equal to 61%. Moreover, Xception classified 2 classes with F1 scores below 90% and the lowest F1 score was equal to 55%. Our proposed combination scheme, the DTEXNet, used the strengths of DT and of the CNN models to classify the attacks resulting in a F1-score of 100% approximately in all classes except for two classes in both datasets. Such performance proves the success of combining machine learning models and CNNs in classification problems generally, and in DDoS attacks classification problems specifically. CNN models were able to extract the important features efficiently from the images which made it easier for the DT to break down the data, make decisions, and classify the attacks regardless of how similar or different these attacks were and regardless of the complexity of the attack's features. The performance of DT baseline model can possibly be enhanced to match that of DTEXNet, however, this will require tremendous efforts to construct feature engineering procedures that extract only the most important and relevant features from the samples in order to enable the DT model to classify the attacks with acceptable results. This step was not necessary in DTEXNet since the CNN models were able to efficiently extract the important features needed to improve the performance of DT with no human effort involved. The excellent performance of DTEXNet is a big advantage of our solution. However, DTEXNet depends on converting the data samples into images, which requires sufficient computational resources and may cause latency issues. In addition, the data conversion may cause some delays if the solution is tested on a real-time data stream. The solution is also developed to work only on the given set of features and does not automatically adapt to new features or new attack types. These are the limitations we identify in our proposed solution and that we plan to explore as future research directions.

5. Conclusions and Future Work

In this paper we propose a novel algorithm for intrusion detection, the DTEXNet, and show its advantages in this setting. Our solution involves training a classical Machine Learning model, using the original feature set concatenated with the features extracted from EfficientNet and Xception. Our DTEXNet Algorithm outperforms each of the three individual models when applied independently showing an excellent performance in the intrusion detection domain. Using DTEXNet expands the room for enhancements and improvements for any tabular data machine learning applications. Converting the structured data into image-like shapes permits harnessing the feature extraction capabilities of the

CNNs. Moreover, the expert knowledge needed to extract features from the raw data to be then fed to the ML model is not as crucially required as before, since the introduced CNN automatically extracts the most important features, and allows a large room for parameter tuning to tailor the extraction capabilities to the dataset. The proposed DTEXNet solution requires sufficient computational resources for training the model, however, the inference speed is no slower than the conventional methods. In addition, the adaptive capabilities of the model do not need periodic retraining on a whole modified dataset, incremental training would suffice. The exploration of effective implementations of our proposed solution that allow it to efficiently handle real-time data streams, adapting to changing features and new types of DDoS attacks is a promising future research avenue.

References

[1] M. Islam, M. Chowdhury, H. Li, and H. Hu. "Cybersecurity attacks in vehicle-to-infrastructure applications and their prevention". In: *Transportation research record* 2672.19 (2018), pp. 66–78.

[2] M. Semerci, A. T. Cemgil, and B. Sankur. "An intelligent cyber security system against DDoS attacks in SIP networks". In: *Computer Networks* 136 (2018), pp. 137–154.

[3] F. Hussain, S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, and G. A. Shah. "IoT DoS and DDoS Attack Detection using ResNet". In: *23rd INMIC*. IEEE. 2020, pp. 1–6.

[4] S Lakshminarasimman, S Ruswin, and K Sundarakantham. "Detecting DDoS attacks using decision tree algorithm". In: *4th ICSCN*. IEEE. 2017, pp. 1–6.

[5] G. Lucky, F. Jjunju, and A. Marshall. "A lightweight decision-tree algorithm for detecting DDoS flooding attacks". In: *20th QRS-C*. IEEE. 2020, pp. 382–389.

[6] H. Wang and W. Li. "DDosTC: A transformer-based network attack detection hybrid mechanism in SDN". In: *Sensors* 21.15 (2021), p. 5047.

[7] I. Mihai-Gabriel and P. Victor-Valeriu. "Achieving DDoS resiliency in a software defined network by intelligent risk assessment based on neural networks and danger theory". In: *15th CINTI*. IEEE. 2014, pp. 319–324.

[8] R. Kokila, S. T. Selvi, and K. Govindarajan. "DDoS detection and analysis in SDN-based environment using support vector machine classifier". In: *6th ICoAC*. IEEE. 2014, pp. 205–210.

[9] T. V. Phan, T. Van Toan, D. Van Tuyen, T. T. Huong, and N. H. Thanh. "OpenFlowSIA: An optimized protection scheme for software-defined networks from flooding attacks". In: *Sixth ICCE*. IEEE. 2016, pp. 13–18.

[10] T. V. Phan, N. K. Bao, and M. Park. "A novel hybrid flow-based handler with DDoS attacks in software-defined networking". In: *UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld*. IEEE. 2016, pp. 350–357.

[11] S. Sarraf et al. "Analysis and detection of ddos attacks using machine learning techniques". In: *ASRJETS* 66.1 (2020), pp. 95–104.

[12] M. D. Prasad, V. P. Babu, and C Amarnath. "Machine learning ddos detection using stochastic gradient boosting". In: *Int. J. Comput. Sci. Eng* 7.4 (2019), pp. 157–16.

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

[14] M. Tan and Q. Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6105–6114.

[15] F. Chollet. "Xception: Deep learning with depthwise separable convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1251–1258.

[16] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani. "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization". In: *4th IICISSP*. 2018.

[17] P. Branco, L. Torgo, and R. P. Ribeiro. "A survey of predictive modeling on imbalanced domains". In: *ACM Computing Surveys (CSUR)* 49.2 (2016), pp. 1–50.

[18] J.-G. Gaudreault, P. Branco, and J. Gama. "An Analysis of Performance Metrics for Imbalanced Classification". In: *International Conference on Discovery Science*. Springer. 2021, pp. 67–77.