Performance Evaluation of Botnet Detection using Deep Learning Techniques

Beny Nugraha
Chair of Communication Networks
Technische Universität Chemnitz
Chemnitz, Germany
beny.nugraha@etit.tu-chemnitz.de

Anshitha Nambiar
Chair of Communication Networks
Technische Universität Chemnitz
Chemnitz, Germany
anshitha.nambiar@s2017.tu-chemnitz.de

Thomas Bauschert
Chair of Communication Networks
Technische Universität Chemnitz
Chemnitz, Germany
thomas.bauschert@etit.tu-chemnitz.de

***Abstract*—Botnets are one of the major threats on the Internet. They are used for malicious activities to compromise the basic network security goals, namely Confidentiality, Integrity, and Availability. For reliable botnet detection and defense, deep learning-based approaches were recently proposed. In this paper, four different deep learning models, namely Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), hybrid CNN-LSTM, and Multi-layer Perception (MLP) are applied for botnet detection and simulation studies are carried out using the CTU-13 botnet traffic dataset. We use several performance metrics such as accuracy, sensitivity, specificity, precision, and F1 score to evaluate the performance of each model on classifying both known and unknown (zero-day) botnet traffic patterns. The results show that our deep learning models can accurately and reliably detect both known and unknown botnet traffic, and show better performance than other deep learning models.**

Keywords—Botnet Traffic Detection, CTU-13 Dataset, Deep Learning, Neural Network, Performance Evaluation, Zero-Day Attack

I. INTRODUCTION

The increase in the complexity of the Internet architecture comes along with threat scenarios that are constantly changing. Attackers are always trying to find a way to exploit vulnerabilities that can exist in a broad number of areas, including devices, data, applications, users, and locations. One of the major threats is malicious botnets. They usually comprise three components: the main attacker known as the botmaster, one or multiple infected machines called bot(s), and a command-and-control (C&C) server. Common botnet communication processes are as follows: First, the botmaster remotely controls the botnet by sending instructions to the C&C server or directly to the bots in the network. Afterward, the controlled bots perform malicious tasks by executing the commands [1]. Botnets pose a significant and growing threat compromising three basic principles of network security - Confidentiality, Integrity, and Availability. For instance, botnets are capable of launching Distributed Denial of Service (DDoS) attacks [2], which can be used to compromise network and service availability.

Host-based and network-based approaches are the two most common botnet detection techniques. The first technique identifies abnormal computation resource usage, for instance, it monitors an abnormal increase of CPU and memory utilization. The later technique analyzes the network and traffic conditions during the bot lifecycle [3]. The advantage of the host-based technique is that it can be applied despite encrypted communication. However, this technique requires more effort because it monitors the resource utilization of all end hosts for a long time. The network-based approach can be further classified into signature-based and anomaly-based. A signature-based method applies Deep Packet Inspection (DPI) on IP packets. Its advantage is the low false positive rate. It is mainly used for detecting known botnets. The disadvantage is that, in order to detect new attack patterns, the signatures need to be updated frequently. Moreover, the signatures can be hidden by using encryption techniques [4]. Anomaly-based methods detect abnormalities based on parameters such as packet payload size and bots behavior. However, the botnet attack detection is more difficult in case the botnet behavior frequently changes over time [5]. Machine learning techniques emerged to be utilized in anomaly-based methods because they perform well in detecting anomalous traffic patterns [6], [7]. However, anomaly-based methods, in general, yield a high false positive rate, leading to many detection errors [4]. Moreover, conventional machine learning techniques pose one major disadvantage, i.e. they rely on a feature engineering process that usually involves human intervention and takes a lot of effort [8]. Because of the above limitations, neural network-based deep learning approaches for network security applications were recently proposed and became a hot research topic. With deep learning, the selection of suitable features (out of all features) is performed autonomously. Therefore, deep learning techniques are well suited to cope with a dataset with a large number of features.

In this paper, we provide a performance analysis of four different deep learning models, namely Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), hybrid CNN-LSTM, and Multi-Layer Perception (MLP) applied for detecting known and unknown botnet traffic patterns. We train, validate, and test the models using the CTU-13 dataset, which is a real botnet traffic dataset that was generated in the Czech Republic at the CTU University in 2011 [9]. Moreover, the effect of having unbalanced traffic data (i.e., a large difference in the amount of benign and botnet traffic) is analyzed in this paper. Various performance metrics, namely Accuracy, Sensitivity, Specificity, Precision, and F1 score are considered. Lastly, a performance comparison between our deep learning models and standard machine learning techniques is presented.

The remaining of the paper is organized as follows: Section II outlines previous work related to deep learning-based botnet detection. Section III describes the CTU-13 dataset, the dataset pre-processing, and feature selection. Section IV describes the deep learning models and the performance metrics. Section V provides and analyzes the obtained performance evaluation results. Section VI concludes the paper summarizing our main findings and outlining ideas for future work.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

978-1-7281-8055-7/20/$31.00 ©2020 IEEE

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

II. Related Work

In this section, we provide a survey of existing approaches for deep learning-based botnet detection.

Maeda et al. [3] proposed a deep learning-based method based on a Multilayer Perceptron (MLP) model to extract a specific set of features for each session from the traffic data and to separate the infected machine after the detection. They use the same botnet dataset (CTU-13 dataset) like in our paper for evaluation - however, it is not specified which of the 13 scenarios of the CTU-13 dataset are applied. Furthermore, they claimed that the MLP model performs best, but they did not compare it with other models. In our work, four different deep learning models are evaluated using all scenarios in the CTU-13 dataset.

Chen et al. [7] presented a CNN-based botnet detection system that is effective in identifying Peer-to-Peer (P2P) botnets. Their system might be able to identify botnets from well-known P2P botnet datasets with high detection rates and low false positive rates. However, they did not consider other performance metrics. In our work, besides Accuracy, several other performance metrics such as Precision, Sensitivity, Specificity, and F1 score are used.

Bontemps et al. [10] followed a collective anomaly detection approach by using the Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) model. Their results show that it is possible to obtain a 100% detection rate, but with generating a high amount of false alarms. Other performance metrics were not considered.

Roopak et al. [11] implemented four different deep learning models using the CICIDS2017 dataset and compared their performance to machine learning algorithms. According to their results, the MLP model performed worst while the CNN-LSTM model performed best. However, they did not provide any explanation of their results.

Maimo et al. [12] propose a novel two-level deep machine learning model where the first level comprises a supervised or semi-supervised learning method based on a Deep Belief Network (DBN) or a Sparse Autoencoder (SAE). The collected anomaly symptoms are passed on to a Network Anomaly Detection (NAD) component, where they are assembled and used as input for an LSTM-RNN. Two different training, validation, and testing schemes were applied. In the first scheme, to classify known botnet traffic, all scenarios in the CTU-13 dataset are combined, then a splitting into training, validation, and testing data is performed. For this scheme, a recall of 99.34% was achieved (i.e., botnet traffic was correctly identified by 99.34%). However, the Precision was not convincing as 18.74% of benign traffic was classified as botnet traffic. Regarding the second scheme, for classifying unknown botnet traffic, scenarios 3, 4, 5, 7, 10, 11, 12, and 13 were used for training and validation, whereas scenarios 1, 2, 6, 8, and 9 were used for testing. The results obtained for this scheme were quite worse as on average only 71% of the botnet traffic was correctly identified.

The CTU-13 dataset was also used in a more recent study by Ahmed et al. [13]. They utilized the feed-forward backpropagation ANN technique to detect zero-day botnet attack traffic and achieved 99.6% Accuracy. However, Accuracy was the only performance metric that they used. This metric should not be used as the only one in a prediction model because it is prone to the so-called Accuracy paradox [14], meaning that a high Accuracy does not necessarily imply a good performance (i.e., detecting anomalous traffic). Moreover, the authors used only a small amount of data from the CTU-13 dataset (8000 traffic flows for training and 2000 traffic flows for testing) and did not specify how they partition the dataset for training and testing (i.e., which scenarios are taken for training and testing). Therefore, for detecting zero-day botnet traffic, they might have used data from different scenarios for training and testing, respectively.

In our work, we apply and thoroughly compare four different deep learning models, namely CNN, LSTM, hybrid CNN-LSTM, and MLP. These models are considered because they have been previously utilized for anomaly detection and obtained very promising results. For instance, the CNN-based detection model was utilized by Chen et al. [7] and Wang et al. [15] for P2P botnet detection and malware traffic classification, respectively. The detection Accuracy turned out to be 98.6% and 99.41%, respectively. Tran et al. [16] proposed to use the LSTM model for botnet detection and they stated that the LSTM model is robust against datasets with unbalanced data. Furthermore, the work by Loukas et al. [17] showed that the LSTM model provides high Accuracy when being applied to unknown data sets. These two aspects (robustness w.r.t. unbalanced data and high Accuracy in case of unknown data) motivated us to consider the LSTM model as well. Among others, a hybrid CNN-LSTM model was applied by Roopak et al [11] to detect DDoS attacks using the CICIDS2017 dataset. Their results showed that the hybrid CNN-LSTM model yields the best performance providing 97.16% Accuracy, 97.41% Precision, and 99.1% Sensitivity. Ferrag et al. [18] investigated the use of an MLP model for anomaly detection taking the CSE-CIC-IDS2018 and the Bot-IoT dataset. Their MLP model achieved more than 96% Accuracy. To extend the performance analysis compared to the mentioned previous studies, we consider two different use cases and apply various performance metrics such as Accuracy, Precision, Sensitivity, Specificity, and F1 score. The two use cases are as follows:

- All CTU-13 scenarios are combined and used for training and testing to evaluate the performance of each deep learning model on classifying traffic from known botnets.
- Different CTU-13 scenarios are used for training and testing to evaluate the performance of each deep learning model on classifying traffic from unknown (zero-day) botnets.

In addition to the above use cases, we train and test our deep learning models for each scenario of the CTU-13 dataset to analyze the impact of unbalanced data. Lastly, we compare the performance of our deep learning models with standard machine learning techniques such as J48 Tree Classifier and Random Forest.

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

TABLE I. Botnet attack scenarios of the CTU-13 dataset

Scenario	Attack Type	Bot Type	Number of Bots
1	IRC, SPAM, CF	Neris	1
2	IRC, SPAM, CF	Neris	1
3	IRC, PS, Custom	Rbot	1
4	IRC, DDoS, Custom	Rbot	1
5	SPAM, PS, HTTP	Virut	1
6	PS	Menti	1
7	HTTP	Sogou	1
8	PS	Murlo	1
9	IRC, SPAM, CF, PS	Neris	10
10	IRC, DDoS, Custom	Rbot	10
11	IRC, DDoS, Custom	Rbot	3
12	P2P	NSIS.ay	3
13	SPAM, PS, HTTP	Virut	1

TABLE II. Number of traffic flows for each scenario of the CTU-13 dataset

Scenario	Number of botnet flows	Number of benign flows
1	2693	8839
2	14362	5267
3	24	27433
4	1931	23731
5	901	4679
6	4630	7494
7	63	1677
8	1520	36625
9	8686	16690
10	74907	13052
11	8164	2718
12	2168	7628
13	23779	13199

III. CTU-13 Dataset, Dataset Pre-Processing, and Feature Selection

A. CTU-13 Dataset

For the evaluation of the deep learning models, we use the CTU-13 dataset which is a botnet traffic dataset created in the Czech Republic at CTU University in 2011 [9]. The data in the CTU-13 dataset is stored in a matrix format and comprises 13 scenarios from various types of botnets [9]. Despite derived in 2011, it is still relevant for current botnet research because each scenario contains detailed traffic traces from a particular botnet and for different types of attack (e.g. IRC, P2P, HTTP attacks, SPAM messages, Click-Fraud (CF), port-scan (PS), DDoS attacks, custom attacks compiled by the authors). As the CTU-13 dataset exclusively comprises botnet traffic traces it is better suited for botnet research than more recent datasets such as CSE-CIC-IDS2018 [19]. Compared to the CSE-CIC-IDS2018 dataset, the CTU-13 dataset includes more attack types and the botnet traffic stems from one or more infected hosts (bots). Each bot uses either the Neris, Rbot, Virut, Menti, Sogou, Murlo, or NSIS.ay botnet tool, while in the case of the CSE-CIC-IDS2018 dataset only the Ares tool is used and only one bot performs the attack. Table I shows for each scenario details of the botnet attack types as well as the type and number of bots. The number of botnet and benign traffic flows for each scenario is presented in Table II – this information is publicly available [20].

The CTU-13 dataset is a dataset in which each traffic flow is labeled as attack, benign, or background traffic. The labeling is performed as follows: attack labels are assigned to all traffic flows between infected systems (bots) and victim systems (attack targets), benign labels are assigned to all other traffic flows from or to the victims, and background labels are assigned to traffic flows not affecting the victims. Consequently, we skip the background traffic flows from the CTU-13 dataset, as they are not relevant for our study.

B. Feature Selection and Dataset Pre-Processing

The following features are extracted from the CTU-13 dataset: traffic flow start time, duration, protocol, source address, source port, direction, destination address, destination port, state, type of service (TOS), total packets, total bytes, and source bytes. In addition, we add two more features that in our view are relevant to differentiate botnet traffic from benign traffic, namely the average byte rate and average packet rate. Abnormal values of these features are indicators for botnet traffic entering the network. The values corresponding to each feature lie in quite different ranges. As the largest values might dominate the smallest ones, scaling is performed so that all values are in a pre-defined range. In the next step, the data is split into training and validation as well as testing data. The training data is used to learn the traffic patterns and the testing data is used to evaluate how well the prediction of the respective deep learning model works in a real-world scenario. The validation data is used to check at every learning round (i.e. after the training is finished) whether the model is trained correctly. In addition to the aforementioned features, we add label "0" and label "1" for benign and botnet traffic, respectively, to perform binary classification.

IV. Deep Learning-based Botnet Detection

In this section, we first outline the four deep learning models that were tailored and applied for botnet detection (remark: these models are available in GitHub [21]). The corresponding hyperparameters (such as learning rate, CNN filter and kernel size, type of activation function, and the number of epochs) are tuned so as to achieve optimal performance. We refer to [22] for a more detailed explanation about hyperparameter tuning.

Afterward, we describe the metrics that are used in our performance analysis, and finally, we present and discuss the obtained results.

A. CNN Model

Fig. 1 (a) illustrates the structure of the implemented CNN model. The indices $x_1 - x_n$ denote the features associated with a traffic flow, i.e. each flow belonging to a scenario is fed into the model as {feature 1, feature 2, … , feature n, and label}. The label is fed to the model as well because we consider supervised learning. This format is the same for all four deep learning models. The first two layers of the CNN model are 1D-CNN layers. A 1D-CNN layer is used instead of a 2D-CNN layer because it allows a faster training and does not require a dedicated Graphical Processing Unit (GPU) to run [23]. In our CNN model, both 1D-CNN layers have 8 filters and a kernel size of 2. The dropout layer (with rate 0.5) is placed after the two 1D-CNN layers to overcome overfitting. The pooling layer is used to remove features with low scores (weights) and keep features with the highest score. A fully connected layer is placed after the pooling layer. This layer is proven to be able to generate a higher-order feature representation to easily classify different classes [24]. Therefore, a fully connected layer is also implemented in our other deep learning models. The

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

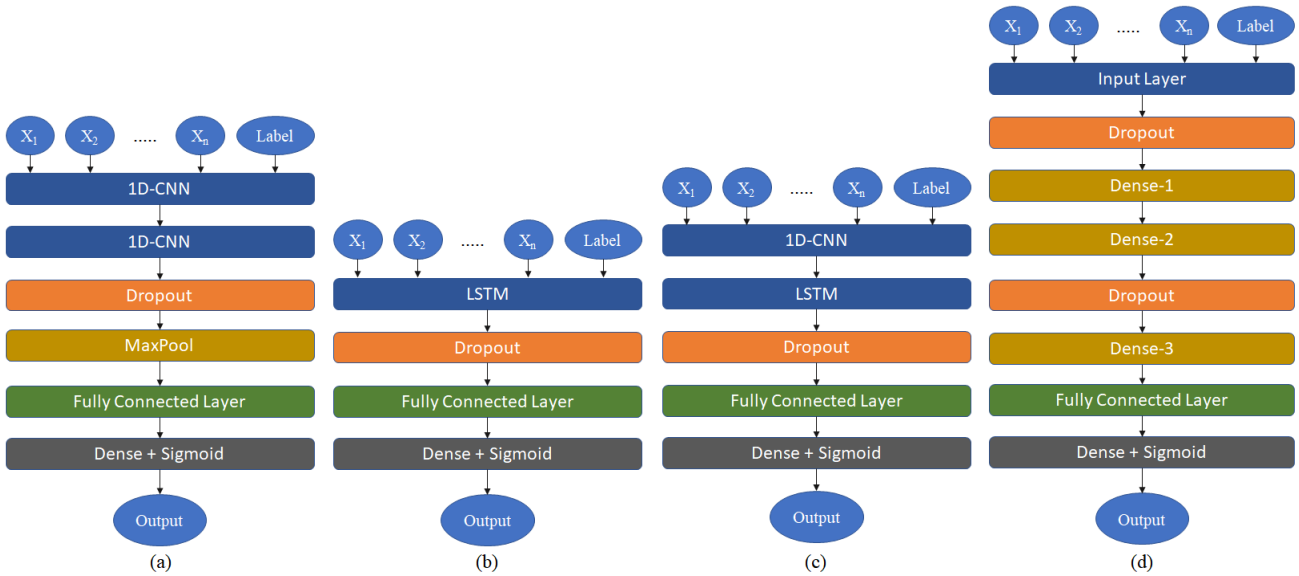

Fig. 1. Deep learning models: (a) CNN model, (b) LSTM model, (c) hybrid CNN-LSTM model, and (d) MLP model

Rectified Linear Unit (ReLU) activation function is applied in the 1D-CNN layers and the fully connected layer. It is used, because according to [25] it shows better performance (wrt. learning time and performance metrics) than other activation functions, e.g. the tanh function. The ReLU function returns value 0 when it receives a negative input and returns the value as it is when it receives a positive input. The output from the fully connected layer is then fed to the last layer which is the dense layer with sigmoid activation function. This layer is used to finally perform the classification into attack and benign flows.

B. *LSTM Model*

Fig. 1 (b) shows the architecture of an LSTM model which is a type of RNN. The LSTM has an internal memory that enables it to accumulate and save information from previous inputs when reading the input data. Furthermore, it is able to handle multiple features [26]. The LSTM model in our implementation first comprises the LSTM layer with 128 neurons using ReLU activation. The number of neurons is chosen so that it is neither too small for proper learning nor too large for properly handling the testing data. The LSTM layer is followed by a dropout layer with a rate of 0.5 in order to avoid overfitting. The output from the dropout layer is fed into a fully connected layer which then is connected to the last dense layer with one neuron and sigmoid activation function.

C. *Hybrid CNN-LSTM Model*

The CNN-LSTM model is composed of a CNN and an LSTM layer as illustrated in Fig. 1 (c). Similar to the CNN model, a 1D-CNN (with 8 filters and a kernel size of 2) is used as the first layer. The second layer comprises the LSTM layer with 128 neurons using ReLU activation. The dropout layer with a rate of 0.5 is placed before the fully connected layer and the dense layer with sigmoid activation function.

D. *MLP Model*

Fig. 1 (d) depicts the structure of the implemented MLP model. The MLP model consists of an input layer with ReLU activation followed by a dropout layer with a rate of 0.5. The dropout layer is followed by two dense layers and a further dropout layer with a rate of 0.5. The output from this dropout layer is then fed into a dense layer and then to a fully connected layer. The last layer is a dense layer with a sigmoid activation function.

Predicted Labels

True Labels		**Botnet**	**Normal**
	Botnet	True Positive (TP)	False Negative (FN)
	Normal	False Positive (FP)	True Negative (TN)

Fig. 2. Confusion matrix

The fully connected layer that is utilized in all models provides a buffer between the learned features and the output. This helps in interpreting the learned features before making a prediction. All models have a dense output layer with a sigmoid activation function because the classification decision is binary (benign or attack flow). In addition to that, Adam optimizers with a learning rate of 0.001 are used in all four models [27], [28]. As mentioned by the authors of Adam, this method has many benefits compared to other optimization algorithms. It is a simple and computationally efficient method aimed to deep learning scenarios dealing with large input datasets such as the CTU-13 dataset.

E. *Performance Metrics*

The performance of the four deep learning models concerning the detection of Botnet attacks is evaluated by using the metrics Accuracy, Sensitivity, Specificity, Precision, and F1 Score [29]. The entries of the confusion matrix that are used to calculate the performance metrics are depicted in Fig. 2.

The metric Accuracy is related to classification Accuracy. It is defined as the ratio of the number of correct predictions and the total number of input samples:

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

The ratio of actual positive cases that are predicted as positive (true positive) and the sum of true positives and false negatives is called Sensitivity (or Recall):

$$Sensitivity\ (Recall) = \frac{TP}{(TP+FN)} \quad (2)$$

The increase in true positives and the decrease in false negatives implies a higher value of Sensitivity. The decrease of true positives and the increase of false negatives reduces the value of Sensitivity. Models that yield high Sensitivity are most desired.

The ratio of actual negative cases (benign flows) that are predicted as negative (true negative) and the sum of true negatives and false positives is called Specificity:

$$Specificity = \frac{TN}{(TN+FP)} \quad (3)$$

Precision is the ratio of correctly predicted botnet flows (true positives) and the total predicted botnet flows (true positives and false positives):

$$Precision = \frac{TP}{(TP+FP)} \quad (4)$$

High Precision implies a low false positive rate, therefore stating how well the model correctly predicts the actual botnet flows out of all predicted botnet flows.

F1 Score is defined as the harmonic mean of Sensitivity and Precision:

$$F1\ Score = 2 * \left(\frac{Sensitivity*Precision}{Sensitivity+Precision}\right) \quad (5)$$

The F1 Score is always closer to the smaller value of Sensitivity or Precision. It is a valuable metric if there is a large difference between Sensitivity and Precision. For instance, in case of high Sensitivity and low Precision, the model yields a low number of false negatives but a high number of false positives.

V. Performance Evaluation Results and Analysis

As mentioned in [30], learning curves are an effective tool to determine whether a model is underfitting, overfitting, or fits well for the considered scenario. In our case, we apply the learning curves to illustrate and compare the prediction error of a model vs. the epoch count in case it is fed with training data ("training error") and validation data ("validation error"), respectively. For a detailed explanation of how to calculate the prediction error see [31]. The model is considered to perform well if both training and validation errors are close to zero. The examples of the learning curves are shown in Fig. 3. Underfitting occurs when the training and validation errors remain constant or decrease but are never close to zero (see Fig. 3 (a)), while overfitting occurs when the validation error first decreases and then increases again (see Fig. 3 (b)). The use of a dropout layer ensures that both the training and validation error decrease and the gap between them is small which is an indicator for a good fit (see Fig. 3(c)). For all evaluation scenarios, we set the number of epochs to 50.

A. Use Case 1: Classifying Traffic from known Botnet Attacks

In the first use case, we evaluate the performance of our models considering traffic from known botnets. We combine the data of all 13 scenarios and split them into training and validation data (80% of total data), and testing data (20% of total data). The split ratio of training data and validation data is also 80:20.

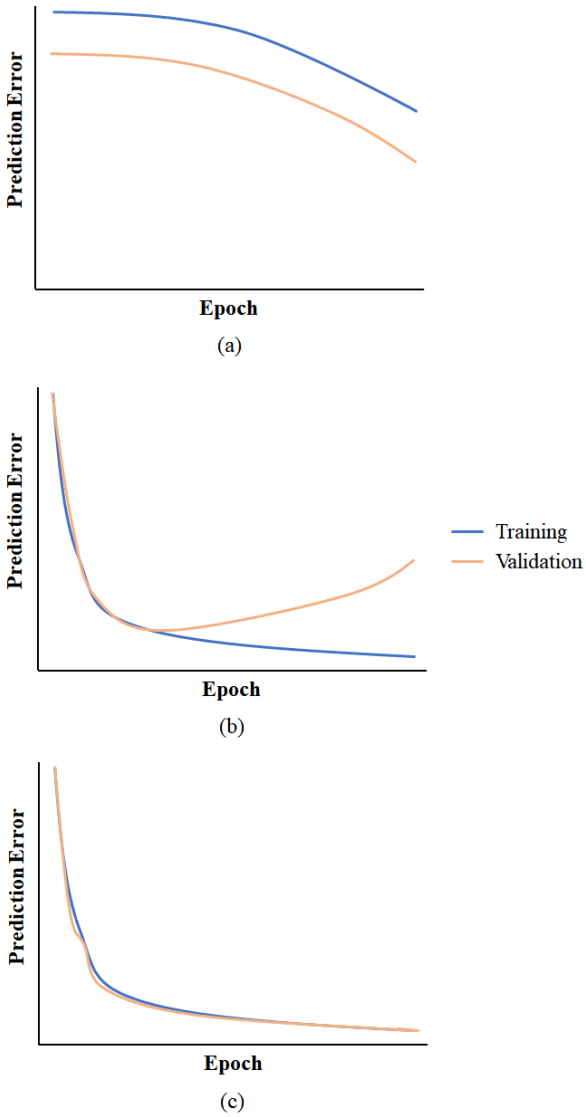

Fig. 3. Learning curve examples: (a) Underfitting, (b) Overfitting, and (c) Good fit

From the results that are presented in Fig. 4 we can conclude that CNN is the best performing model. It is able to correctly identify the botnet traffic flows by 99.587%. The resulting Accuracy, Specificity, Precision, and F1 Score are quite good with 99.001%, 99.440%, 99.467%, and 99.025%, respectively.

Performance comparison of our CNN model and the model from Maimo et al. (first scheme) [12] is presented in Table III. It can be seen, that regarding the identification of benign traffic flows, our CNN model outperforms Maimo et al.'s model, and in addition, the F1 Score is better as well. However, Maimo et al.'s model outperforms our model regarding the identification of the botnet traffic flows. Unfortunately, the two models cannot be compared in more detail because Maimo et al. did not provide Accuracy and Specificity values.

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

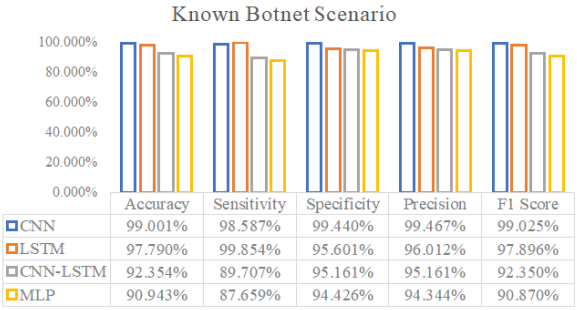

Fig. 4. Results for use case 1

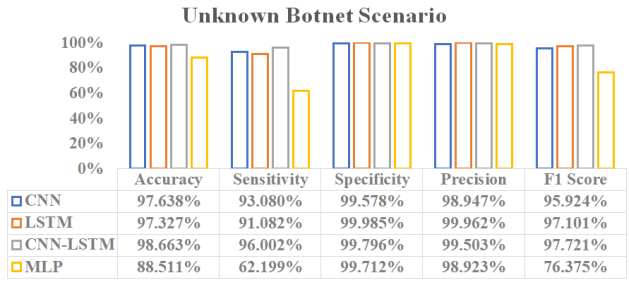

Fig. 5. Results for use case 2

TABLE III. Use case 1: Comparison of our results with other work

	Accuracy	Sensitivity	Specificity	Precision	F1 Score
[12]	-	99.3%	-	81.3%	89.4%
Our Work	99.001%	98.587%	99.440%	99.467%	99.025%

TABLE IV. Use case 2: Comparison of our results with other work

	Accuracy	Sensitivity	Specificity	Precision	F1 Score
[12]	-	70.9%	-	68.6%	68.3%
[13]	99.6%	-	-	-	-
[32]	89.4%	85.7	93.1%	92.5%	88.9%
Our Work	98.6%	96%	99.8%	99.5%	97.7%

TABLE V. Use Case 3: Results for Scenario 3 and Scenario 6

		Scenario 3	Scenario 6
CNN	Accuracy	99.985%	99.992%
	Sensitivity	83.333%	99.978%
	Specificity	100.000%	100.000%
	Precision	100.000%	100.000%
	F1 Score	90.909%	99.989%
LSTM	Accuracy	99.913%	99.975%
	Sensitivity	0.000%	99.978%
	Specificity	100.000%	99.947%
	Precision	Unmeasurable	99.957%
	F1 Score	Unmeasurable	99.968%
CNN-LSTM	Accuracy	100.000%	100.000%
	Sensitivity	100.000%	100.000%
	Specificity	100.000%	100.000%
	Precision	100.000%	100.000%
	F1 Score	100.000%	100.000%
MLP	Accuracy	100.000%	100.000%
	Sensitivity	100.000%	100.000%
	Specificity	100.000%	100.000%
	Precision	100.000%	100.000%
	F1 Score	100.000%	100.000%

B. Use Case 2: Classifying Traffic from unknown (Zero-Day) Botnet Attacks

In order to evaluate the performance of our models on classifying traffic from zero-day botnet attacks, the datasets of the CTU-13 scenarios are partitioned in a way that the training and validation dataset is different from the testing dataset. The authors of CTU-13 recommend to perform the partitioning as follows: the data from scenarios 3, 4, 5, 7, 10, 11, 12, and 13 are used for training and validation, whereas the data from scenarios 1, 2, 6, 8, and 9 are used as testing dataset [9]. The split ratio of training data and validation data remains 80:20. The performance evaluation results of use case 2 are presented in Fig. 5. It can be seen that all our models perform very well in classifying traffic flows from zero-day botnet attacks. The LSTM and the hybrid CNN-LSTM models perform better than the other two models. The LSTM model is able to identify around 91.1% of the unknown botnet traffic flows correctly, while the hybrid CNN-LSTM model is able to identify around 96% of the unknown botnet traffic flows correctly. Moreover, we observe that the execution time of the hybrid CNN-LSTM model is around 81.1% lower compared to the LSTM model. Comparing the results obtained for use case 1 and use case 2, we recognize that the best model in use case 1 (the CNN model) performs better than the best model in use case 2 (the hybrid CNN-LSTM model).

Performance comparison of our hybrid CNN-LSTM model with the models from Maimo et al. (second scheme) [12] , Ahmed et al. [13], and Bansal [32] is presented in Table IV. A detailed comparison of our model with Maimo et al.'s and Ahmed et al.'s model is not possible because they reported fewer performance metrics. The Sensitivity, Precision, and F1 Score of our model are significantly higher compared to Maimo et al.'s model. The Accuracy of Ahmed et al.'s model is slightly higher than in our model. However, as already mentioned, Accuracy should not be used as the only performance metric due to the Accuracy paradox. Our model outperforms also Bansal's model w.r.t. all performance metrics.

C. Use Case 3: Analyzing the Impact of unbalanced Data

From Table II it can be seen that for some scenarios in the CTU-13 dataset there is an unbalanced number of botnet and benign traffic flows. Therefore, in order to analyze the impact of this unbalanced data on the classification performance of our models, the four deep learning models were applied to each of the 13 scenarios of the CTU-13 dataset individually. Here we present and discuss the results of only two scenarios, namely scenario 6 and scenario 3, which are the most balanced and the most unbalanced scenario, respectively. The results for both scenarios are presented in Table V. We first discuss the results of scenario 6. It can be seen that all our models achieve 99%-100% for all metrics. These good results are achieved not only because the number of benign and botnet traffic flows are balanced, but also because the models are trained and tested using the same set of data, i.e. the training and testing data were derived from scenario 6. For the hybrid CNN-LSTM and MLP model, all metrics are even 100% as these models correctly identify all the benign and attack traffic flows (i.e., the number of false positives and false negatives is zero). Next, we discuss the results of scenario 3. For LSTM, the Accuracy is 99.913%, however, the Sensitivity is 0% and both Precision and F1 Score are unmeasurable. This proves the occurrence of the Accuracy paradox. In the following, we analyze the reason for Precision and F1 Score of the LSTM model being unmeasurable. Precision relies on the number of false positives and true positives. As the number of botnet traffic flows in scenario 3 is quite low compared to the number of benign traffic flows, the LSTM model exhibits a

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

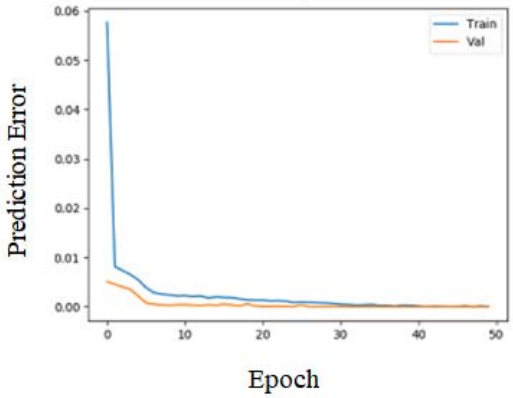

Fig. 6. Training and validation error for the Hybrid CNN-LSTM model applied to Scenario 3

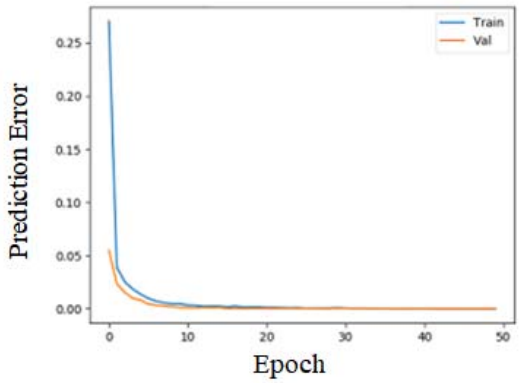

Fig. 7. Training and validation error for the Hybrid CNN-LSTM model applied to Scenario 6

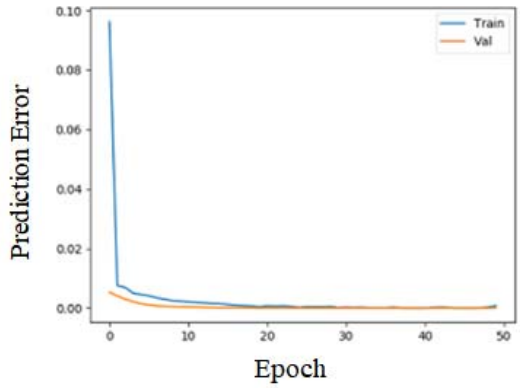

Fig. 8. Training and validation error for the MLP model applied to Scenario 3

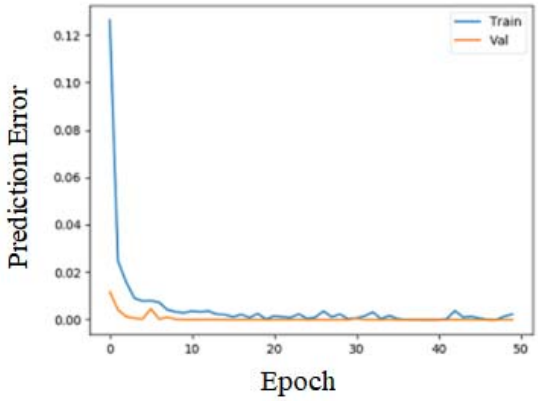

Fig. 9. Training and validation error for the MLP model applied to Scenario 6

bias towards the benign traffic flows, thus, ignoring the botnet traffic flows completely yielding zero false positives and zero true positives. The zero number of true positives is also the reason for the 0% Sensitivity of the LSTM model. The F1 Score is unmeasurable because it relies on Sensitivity and Precision, which are 0% and unmeasurable, respectively. In order to prevent this, one can attempt to reduce the unbalance of benign and botnet traffic flows by either duplicating the botnet traffic flow data or removing some benign traffic flow data [33]. From Table V it can be seen that both the hybrid CNN-LSTM and the MLP model yield 100% for all metrics for both scenario 3 and scenario 6. However, 100% Accuracy can also indicate that the model is overfitting. Therefore, we checked the learning curves for both models. The training and validation error of the hybrid CNN-LSTM model for scenario 3 and scenario 6 is depicted in Fig. 6 and Fig. 7, respectively, while for the MLP model it is illustrated in Fig. 8 and Fig. 9, respectively. From Fig. 6 – Fig. 9, we can conclude that both models do not show any sign of overfitting, thus verifying the results in Table V.

Unbalanced datasets exist for many other applications. Therefore, applying a technique, e.g. data sampling or cost-sensitive learning, to deal with such unbalanced datasets is desirable. Data sampling attempts to balance the number of class samples so that the level of unbalance is reduced. This can be either done by removing random samples from the majority class or by duplicating random samples from the minority class. Cost-sensitive learning attempts to increase the importance of the minority class. This can be performed by considering class weight factors in order to reduce the model's bias toward the majority class or to increase the importance of the minority class [33].

D. Comparison with Machine Learning Techniques

In this sub-section, we compare the performance of our deep learning models to existing machine learning techniques that were investigated by other researchers. We only present the results of previous studies that also used the CTU-13 dataset.

Haq and Singh [34] evaluated the performance of J48 Tree Classifier, K-Means Clustering, and hybrid classification and clustering by using samples of the CTU-13 dataset. Their results show that the best Accuracy that can be achieved is 90.2723% - it is obtained for the J48 Tree Classifier. An exact comparison with our work is not possible as Haq and Singh use a different subset of the CTU-13 dataset. Nevertheless, it is noticeable that their Accuracy results are quite low.

Delplace et al. [35] applied popular machine learning techniques such as Logistic Regression, Support Vector Machine, Random Forest, and Gradient Boosting. They trained the algorithms with 2/3 of the CTU-13 dataset and tested with the remaining 1/3 (randomly chosen). From their investigation, it turns out that the Random Forest technique performs best. The authors further evaluated the Random Forest method for each scenario of the CTU-13 dataset. We compared our hybrid CNN-LSTM model with their Random Forest approach for each scenario of CTU-13 as well - the

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

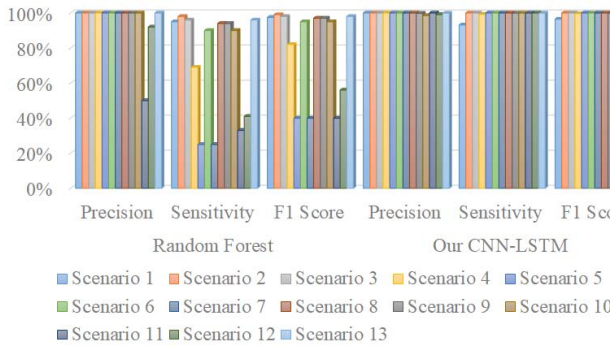

Fig. 10. Comparison of the Hybrid CNN-LSTM model and the Random Forest method

respective results are shown in Fig. 10. Delplace et al. stated that their Random Forest technique performs poor for scenario 4, 5, 7, 11, and 12 because of the small amount of data in the respective scenarios. From Fig. 10 it can be seen that our hybrid CNN-LSTM model is able to perform equally well for all scenarios and that it outperforms the Random Forest technique.

VI. Conclusions

In this paper, the performance of four deep learning models, namely CNN, LSTM, hybrid CNN-LSTM, and MLP applied for botnet attack detection are thoroughly evaluated using the CTU-13 dataset which comprises labeled data of real traffic flows. We focus on the classification of traffic flows stemming both from known and unknown (zero-day) botnet attacks and also discuss the impact of unbalanced data. Regarding the classification of traffic flows from known botnet attacks, all our models achieve good results with the CNN model being the best one. It is able to achieve 100% Specificity and Precision, which means that the model is able to identify 100% of benign traffic flows correctly (i.e. none of the benign traffic flows are classified as botnet traffic flows). For classifying traffic flows from unknown (zero-day) botnet attacks, we observe that the hybrid CNN-LSTM model achieves not only the best results compared to our other models but also compared to the models used in other studies. We also show that the Accuracy paradox can emerge in a scenario with unbalanced data, which leads to the recommendation that other metrics besides Accuracy should also be considered for performance evaluation.

In our future work, we intend to apply and evaluate the presented deep learning models as well as the newer Deep Reinforcement Learning technique in a Software-Defined Networking (SDN) environment for detecting botnet attacks in real-time.

Acknowledgment

This work was performed in the framework of the Celtic-Plus project SENDATE Secure-DCI, funded by the German BMBF (ID 16KIS0481).

References

[1] G. Vormayr, T. Zseby, and J. Fabini, “Botnet Communication Patterns,” In: *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2768-2796, Fourthquarter 2017.

[2] M. Feily, A. Shahrestani, and S. Ramadass. “A survey of botnet and botnet detection”. In: *Third International Conference on Emerging Security Information, Systems, and Technologies*, 2009, pp. 268–273.

[3] S. Maeda, A. Kanai, S. Tanimoto, T. Hatashima, and K. Ohkubo. “A Botnet Detection Method on SDN using Deep Learning”. In: *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019, pp. 1–6.

[4] M. Stevanovic and J. Pedersen, “On the use of machine learning for identifying botnet network traffic,” In: *Journal of Cyber Security and Mobility*, vol. 4, no. 2, pp. 1-32, 2016.

[5] D. Berman, A. Buczak, J. Chavis, and C. Corbett, "A Survey of Deep Learning Methods for Cyber Security,” In: *Information*, vol. 10, no. 4, p. 122, 2019.

[6] G. Kirubavathi and R. Anitha, "Botnet detection via mining of traffic flow characteristics", In: *Computers & Electrical Engineering*, vol. 50, pp. 91-101, 2016.

[7] S.-C. Chen, Y.-R. Chen, and W.-G. Tzeng. “Effective botnet detection through neural networks on convolutional features,” In: *2018 17th IEEE International Conference On Trust, Security, And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 372–378.

[8] M. Najafabadi, F. Villanustre, T. Khoshgoftaar, N. Seliya, R. Wald and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” in *Journal of Big Data*, vol. 2, no. 1, 2015.

[9] S. García, M. Grill, J. Stiborek, and A. Zunino, “An empirical comparison of botnet detection methods,” In: *Computers & Security*, Elsevier. 2014. Vol 45, pp 100-123. http://dx.doi.org/10.1016/j.cose.2014.05.011

[10] L. Bontemps, J. McDermott, N.-A. Le-Khac, et al. “Collective anomaly detection based on long short-term memory recurrent neural networks,” *International Conference on Future Data and Security Engineering*, 2016, Springer, pp. 141–152.

[11] M. Roopak, G. Yun Tian and J. Chambers, “Deep Learning Models for Cyber Security in IoT Networks,” 2019 *IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA, 2019, pp. 0452-0457.

[12] L. Fernandez Maimo, A. Perales Gomez, F. Garcia Clemente, M. Gil Perez, and G. Martinez Perez, “A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks,” *in IEEE Access*, vol. 6, pp. 7700-7712, 2018.

[13] A. Ahmed, W. Jabbar, A. Sadiq and H. Patel, “Deep learning-based classification model for botnet attack detection,” *Journal of Ambient Intelligence and Humanized Computing*, 2020. Available: 10.1007/s12652-020-01848-9.

[14] F. Valverde-Albacete, J. Carrillo-de-Albornoz and C. Peláez-Moreno, “A Proposal for New Evaluation Metrics and Result Visualization Technique for Sentiment Analysis Tasks,” in *International Conference of the Cross-Language Evaluation Forum for European Languages*, Valencia, Spain, 2013.

[15] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye and Yiqiang Sheng, “Malware traffic classification using convolutional neural network for representation learning,” 2017 International Conference on Information Networking (ICOIN), Da Nang, 2017, pp. 712-717, doi: 10.1109/ICOIN.2017.7899588.

[16] D. Tran, H. Mac, V. Tong, H. Tran and L. Nguyen, “A LSTM based framework for handling multiclass imbalance in DGA botnet detection,” in Neurocomputing, vol. 275, pp. 2401-2413, 2018. Available: 10.1016/j.neucom.2017.11.018.

[17] G. Loukas, T. Vuong, R. Heartfield, G. Sakellari, Y. Yoon and D. Gan, “Cloud-Based Cyber-Physical Intrusion Detection for Vehicles Using Deep Learning,” in IEEE Access, vol. 6, pp. 3491-3508, 2018, doi: 10.1109/ACCESS.2017.2782159.

[18] M. Ferrag, L. Maglaras, S. Moschoyiannis and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” in Journal of Information Security and Applications, vol. 50, p. 102419, 2020. Available: 10.1016/j.jisa.2019.102419.

[19] I. Sharafaldin, A. Habibi 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*, 2018, pp. 108–116.

[20] The Stratosphere IPS. The CTU-13 Dataset, A Labeled Dataset with Botnet, Normal and Background traffic. Accessed: 03 February 2020. url: https://www.stratosphereips.org/datasets-ctu13

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.

[21] B. Nugraha, "DLTeamTUC/NetworkSecurity", GitHub, 2020. [Online]. Available: https://github.com/DLTeamTUC/NetworkSecurity.

[22] I. Goodfellow, Y. Bengio and A. Courville, *Deep learning*. Cambridge, Mass: The MIT Press, 2016.

[23] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1D convolutional neural networks and applications: A survey," *arXiv preprint arXiv:1905.03554*, 2019.

[24] T. N. Sainath, O. Vinyals, A. Senior and H. Sak, "Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks," in *2015 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Brisbane, QLD, 2015, pp. 4580-4584.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with DeepConvolutional Neural Networks, " in *Commun. ACM60.6*, pp. 84–90, 2017.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," in *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[27] Keras developers. Keras. Accessed: 27 September 2019. url: https://keras.io/optimizers/.

[28] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization," In: *3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 2015.

[29] D. M. Powers, "Evaluation: from Precision, recall and F-measure to ROC, informedness, markedness and correlation," in *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37-63, 2011.

[30] M. Anzanello and F. Fogliatto, "Learning curve models and applications: Literature review and research directions," In: *International Journal of Industrial Ergonomics*, vol. 41, no. 5, pp. 573-583, 2011.

[31] C. Sammut and G. Webb, *Encyclopedia of Machine Learning*. New York: Springer, 2011.

[32] A. Bansal, and S. Mahapatra, "A comparative analysis of machine learning techniques for botnet detection," in *Proceedings of the 10th International Conference on Security of Information and Networks*, pp. 91-98, 2017.

[33] J. Johnson and T. Khoshgoftaar, "Survey on deep learning with class imbalance," in: *Journal of Big Data*, vol. 6, no. 1, 2019.

[34] S. Haq and Y. Singh, "Botnet Detection using Machine Learning," *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, Solan Himachal Pradesh, India, 2018, pp. 240-245, doi: 10.1109/PDGC.2018.8745912.

[35] A. Delplace, S. Hermoso and K. Anandita, "Cyber Attack Detection thanks to Machine Learning Algorithms," in *Advanced Security Report at the University of Queensland*, May 2019. arXiv:2001.06309

Authorized licensed use limited to: ULAKBIM UASL - CUKUROVA UNIVERSITESI. Downloaded on May 16,2021 at 12:26:54 UTC from IEEE Xplore. Restrictions apply.