ADRRI JOURNAL OF ENGINEERING AND TECHNOLOGY
ESSN-L: 2026-674X
VOL. 6, No. 2 (5), July, 2022-September, 2022
PUBLISHED BY AFRICA DEVELOPMENT AND RESOURCES RESEARCH INSTITUTE

ADRRI JOURNAL OF ENGINEERING AND TECHNOLOGY

ADRRI JOURNALS (www.adrri.org)

***E-ISSN*: 2026-674X VOL. 6, No.2 (5), July, 2022-September, 2022**

Convolutional Neural Network Based Model for Multiclass Botnet Classification in IoT

Mustapha Adamu Mohammed[1], Seth Alornyo[2], Michael Asante[3], Bernard Obo Essah[4]
[1,3]Department of Computer Science, Kwame Nkrumah University of Science and Technology-Kumasi, Ghana.
[1]Email: adamu.mohammed@ktu.edu.gh
[3]Email: masante.csi@knust.edu.gh
[1,2]Department of Computer Science, Koforidua Technical University-Koforidua, Ghana
[2]Email: sabigseth@ktu.edu.gh
[4]Department of Mathematics, ST. Gregory Catholic Senior High School
[4]Email: boessah@st.ug.edu.gh
[1]**Correspondence**: adamu.mohammed@ktu.edu.gh
Available Online: 30th September, 2022
URL: https://journals.adrri.org/index.php/home

Abstract

IoT devices have fundamental security flaws that leave them open to a variety of security threats and attacks, including attacks from botnets. Therefore, creators of botnets continue to take advantage of the security vulnerabilities inherent in IoT devices to control many host devices on networks to launch cyber-attacks on their target systems. The ongoing development of techniques to evade and obfuscate existing detection and security procedures makes it difficult to discover IoT bot vulnerabilities. This study proposes a deep learning method to detect two famous botnet-based attacks: the mirai and Bashlite bots on IoT devices. Our approach implements a 1-dimensional convolutional neural network model (1D-CNN) that is trained on 115 features of real traffic data collected from nine commercial internet of things devices infected by the two mentioned IoT bots to recognize 10 classes of attacks and 1 class of benign traffic. The trained multiclass classification malware detection model was evaluated on 847513 samples, containing 7062606 instances from the N-BaIoT dataset. We further trained two existing models: Plain Feed forward neural network and a popular supervised machine learning classifier, (Logistic Regression) models on the same preprocessed datasets, and compared the classification performances against our proposed model. The experimental results show that our 1D neuron-based model produced a higher prediction in terms of overall classification accuracy over the two models. It was further noted that our model's performance was superior to those of earlier studies on deep learning-based IoT botnet detection.

Keywords: deep neural networks, deep learning; IoT botnets; convolutional neural network; long-short term memory networks; IoT security

[Cite Article as: Mohammed, M. A., Alornyo, S., Asante, M. and Essah, B. O. **(2022). Convolutional Neural Network Based Model for Multiclass Botnet Classification in IoT.** ADRRI Journal of Engineering and Technology, Ghana: Vol. 6, No. 2 (5), Pp. 1-13, E-ISSN: 2026-674X, 30th September, 2022.**]**

Received: (July 8, 2022)

Accepted: (September 30, 2022)

INTRODUCTION

The rapid growth of the internet of things (IoT) devices has led to an increase in the size of computer networks. (Nguyen et al., 2020). Even though the IoT paradigm continues to make people live smarter and more convenient lives, there are many security threats posed by these IoT based devices as well.

Malicious software creators have found it easier to target IoT systems due to the large and heterogeneous collection of devices connected to such systems. Furthermore, these devices have a nonrestrictive tendency in allowing the installation of apps from any possible source (Yerima & Alzaylaee, 2020). Many malicious software applications that have the capability to infect IoT devices and turning same into botnets have surfaced in recent times. Eventually, these IoT bots can form larger botnets and exploited by the botmasters to undertake different kinds of attacks; including distributed denial of service attacks (DDoS), click fraud, phishing attacks, spam attacks and credential stuffing attacks.

Basically, an IoT bot is a type of malware once installed on an IoT device or any other smart device, will take over the device to run automatically, receiving commands and to communicate with command and control servers. They then receive instructions and send information to those servers or botmasters directly (Pieterse & Olivier, 2012),(Amini et al., 2015),(Letteri et al., 2018). Any device that is infected then becomes added to the network of IoT bots and managed by the cyber-attacker.

The security threats posed by IoT bots has become more serious and a major setback to the survival of the internet of things ecosystem. This is particularly worrisome because the increasing use of sophisticated schemes such as obfuscation and encryption mechanisms to hide or erase their traces is making it difficult to detect botnets from normal traffic in IoT systems using traditional signature-based schemes such as intrusion detection systems (IDS) and intrusion prevention systems (IPS). This study suggests a deep learning technique based on the convolutional neural network algorithm to classify various IoT bot classes of Bashlite and Mirai attacks on IoT devices. Deep learning-based models use several layers of a neural network to

learn by themselves and keep evolving to match the dynamic and stealthiness of contemporary IoT bots, particularly mirai and Bashlite.

Our one-dimensional convolutional neural network-based classifier uses 115 real attributes from the N-BaIoT dataset to categorize multiple malwares infected by Mirai and Bashlite botnets. The design of our 1D-CNN based model for multiclass classification of IoT bots is presented and the model evaluated on 847513 traffic data collected from physical IoT devices.

In addition to our proposed model, we also trained two existing deep learning models on the same dataset using python's deep learning framework. Finally, we compare the performance of our model against that of feed forward neural networks and machine learning based models in terms of accuracy and F-1 score.

The remainder of the paper is structured as follows: Section 2 discusses the related works in IoT malware detection. Section 3 presents our proposed model design and an overview of convolutional neural networks, the deep learning algorithm adopted in this study. We present the methodology and the experiments undertaken in section 4. The experimental results are presented in section 5. Finally, we present the conclusion of the study in section 6.

LITERATURE REVIEW

Botnets and malware threats have plagued the internet of things systems in recent times. This phenomenon has challenged researchers in finding lasting and effective detection solutions to these threats. Many studies have used dynamic or static analysis to categorize and detected IoT based malware attacks. Stakhanova et al (Abdul Kadir et al., 2015) studied dynamic analysis with static analysis to discover the relationship between multiple families of botnet attacks. Their study generated a great deal of insight on the characteristics of different malicious attacks and their nomenclature in mobile environment. The authors used the ISCX dataset which contained 1929 instances of 14 different botnets to study how the communication between command-and-control server of botnet operate. This study did not only heighten the interest of botnet attack research, but also made the ISCX data a bench mark dataset for botnet research.

Consequently, Anwar et al., 2016, used MD5 hashes, permissions and related services as attributes to design a static method to detect mobile botnet attacks. The authors created a machine learning-based classifier to detect and categorize botnet activity in mobile devices based on features derived from android-based applications. Alqatawna et al., 2017, built a machine learning model to detect andriod based botnets using permissions from the apps as features. Their model was evaluated on existing machine learning classifiers like decision tress, naive bayes and random forest. Using the same ISCX dataset, the authors conducted their experiments on 3270 instances made up of 1635 normal applications and 1635 botnet applications. The random forest classifiers produced the highest detection accuracy in botnet attacks in android environment.

Machine learning methods have been applied for botnet detection in the studies conducted by Ramachandran et al., 2006, Doshi et al., 2018 and Hoang & Nguyen, 2018 over the years, achieving great successes. However, the heterogeneous processor architectures on internet of things devices

poses a couple of challenges in machine learning based methods to botnet detection in IoT systems Nguyen et al., 2020. By examining their prediction pattern, attackers continue to build code evasion schemes and evade machine learning based detection models.

According to McDermott et al., 2018, deep learning methods have been more effective in detecting botnet activities in IoT environment, given the ever expanding nature of the system. Based on this observation, researchers such as Meidan et al (Meidan et al., 2018) and Jung et al Jung et al., 2020 have used deep learning approaches to detect botnet attacks in their studies. The works of these two researchers is of particular interest in our study as they also used the N-BaIoT dataset used in this study.

The study in HaddadPajouh et al., 2018 used deep recurrent neural networks (DRNNs) to classify IoT bots. The researchers based their study on opcode sequences extracted from executable files of IoT applications. The authors trained their model on 551 ARM datasets and reported a performance of over 98 percentage accuracy.

Also, the study by Azmoodeh et al., 2018, deployed deep neural networks to classify internet of battlefield things malicious software. The authors collected opcodes from graphs based on the operation code graph and used deep eigenspace learning to distinguish malware from legitimate programs. The researchers evaluated their model on a dataset made up 128 malicious apps and 1078 normal apps using the ARM IoT executable file. This model reported over 99 percentage accuracy. However, the use of the operation code sequences is prone to disruption by code variations, rendering models produced by the two studies not robust enough.

In a nutshell, prior studies on IoT botnet classification have explored and used different characteristics of the attack to design detection schemes which have achieved high accuracy. However, there exist some common limitations in the existing studies. Most of the studies have used opcode sequences of the malwares which are low level attributes of the botnets and therefore models designed on these features may not be robust. Secondly, most of the existing studies have built models that perform binary classification of IoT apps into botnets or benign apps. Models to classify multiple botnets types in IoT are limited in literature. In this study, we build a multi-class model using the power of convolutional neural network with ReLU for hidden layers and SoftMax activation for dense layers to detect 11 classes of IoT botnets.

Research Objectives

The goal of this study is to build, train and evaluate a deep neural network that can learn a good mapping of real IoT traffic data to output multi-classification model capable of identifying an incoming traffic that has never been seen before as “Normal Traffic” or “Mirai” based botnet traffic or “Bashlite” based botnet traffic. We sought to choose a model configuration and training configuration that produces the lowest loss and highest accuracy possible for our IoT traffic data. Figure 1 presents our proposed 1-dimensional convolutional neural network with three output labels. In its basic structure, a convolutional layer and pooling layers are building blocks needed to construct a convolutional neural network (Yerima & Alzaylaee, 2020). The feature extraction is

carried out by the convolutional layer whereas the pooling layer is responsible for the dimensionality reduction of the features (LeCun et al., 2015). The fully connected layer does the classification. In this layer are a number of dense layers, depending on the nature of the data. If the dataset is more complex, the number of dense (hidden) layers may be more, giving the network a deeper architecture.

Therefore, the is increased computational complexity if many hidden layers are present in our network. Again, the more complex the network, there is the likelihood to have an overfitting model which would perform poorly in the classification. However, we can employ the Dropout (Srivastava et al., 2014) technique to reduce the possibility of model overfitting.

In constructing our model, we employed the sequential approach and incorporated Batch Normalization after each convolution. Because we are working within a 1D neuron, flattening of the data was avoided before entering the dense layers. Because they are all hidden layers, we employed Rectified Linear Units (ReLU) for activation in the dense layer (i.e., not output layers). We set Dense to 3 in the output layer because we wanted the output to be a vector of 3 values, i.e., $(h_\Theta(x)\varepsilon\mathbb{R}^3)$ all of which were probabilities since we employed SoftMax activation function.

Finally, we compile the model using Stochastic Gradient Descent (SGD), with steps used in the gradient descent (learning rate) of 1.0 and 0.9 momentum. We used categorical cross entropy, which is excellent for multiclass classification problems, to calculate the loss. This is why we needed to convert the output labels into categorical variables: i. e. Normal, Mirai and Bashlite.

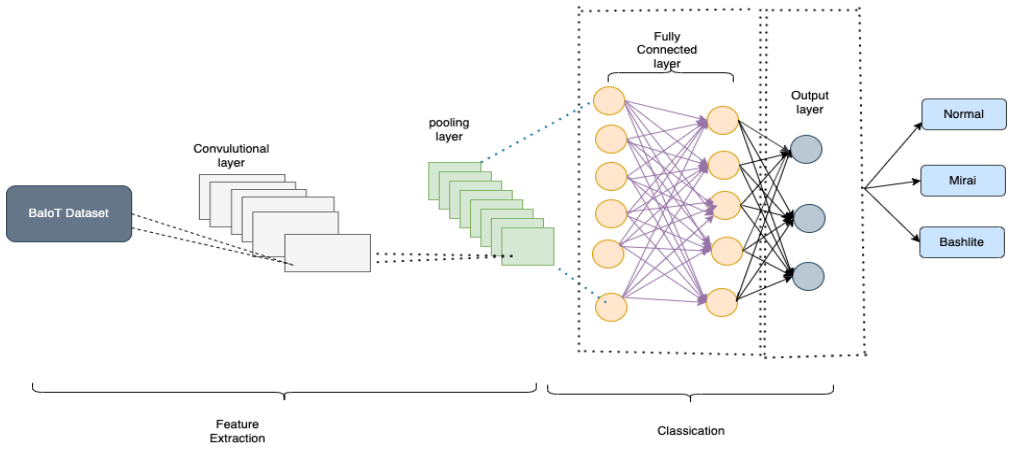

Figure 1: Overview of our proposed 1D neuron with three output units

A multiclass classification problem is where we have more than one category that we try to distinguish. The dataset used in this study is IoT network traffic which comprises benign traffic and malicious traffic launched by mirai and Bashlite botnets. The IoT network is made up of nine devices that are running different applications such as door-bell system, security camera, thermostat, baby monitor etc. Given any received packet by a device in the network, we want to

classify three categories of traffic. Thus, given an IoT traffic, we want to detect whether it is a normal traffic, mirai based attack or bashlite based attack. If that is the case, we will build a 1D neuron with 3 output units. Essentially, our neural network outputs a vector of 3 numbers, represented as $(h_\Theta(x)\varepsilon\mathbb{R}^3)$.

So, we try to get the first output unit to determine "is the packet normal traffic? Yes/No". The second output unit to determine "is the packet mirai based attack"? Yes/No. The third output unit to determine 'is the packet Bashlite based attack? Yes/No.

If the packet is benign traffic, ideally, we will want our network to output: $h_\Theta(x)=\begin{bmatrix}0\\0\\1\end{bmatrix}$, when the packet is bashlite based attack.

So, this is just an extension of one-versus-all multiclass classification technique in a neural network, which is a regression. And here we have essentially three regression classifiers, each trying to recognize one of three class labels that we sought to classify. Therefore, this is our neural network with three output units and those are what we want each unit to have. Consequently, training set: $\left(x^{(1)},y^{(1)}\right),\left(x^{(2)},y^{(2)}\right)$ and $\left(x^{(3)},y^{(3)}\right)$. what we are going to do with the output y where as in the labels being an integer from 1,2,3. $y\varepsilon\{1,2,3\}$. Instead of representing y this way, we rather represent output y as follows:

$y^{(i)}$ will be either one of $\begin{bmatrix}1\\0\\0\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix}$, or $\begin{bmatrix}0\\0\\1\end{bmatrix}, y\varepsilon\{1,2,3\},\left(x^{(i)},y^{(i)}\right),h_{\Theta(2)}\left(x^{(i)}\right)\approx y^{(i)}\equiv\mathbb{R}^3,$ pending on what their corresponding traffic data x, y is. One training, example will be one pair $\left(x^{(i)},y^{(i)}\right)$ where $x^{(i)}$ is traffic within one of the three labels where $y^{(i)}$ will be one of vectors

$\begin{bmatrix}1\\0\\0\end{bmatrix},\ \begin{bmatrix}0\\1\\0\end{bmatrix},\ \begin{bmatrix}0\\0\\1\end{bmatrix}$, and hopefully, we can find a way to get our neural network to output some values, so

$h_\Theta\left(x^{(i)}\right)\approx y^{(i)}$ and both $h_\Theta\left(x^{(i)}\right)$ and $y^{(i)}$ are going to be 3 dimensional vectors 3 when we have a 3 class labels to classify.

METHODOLOGY

We built our neural network model in python 3.9 using Keras 2.9.0 API on TensorFlow version 2.5.0. as backend. Figure 2 shows the summary of the architectural overview of our proposed neural network. The essential components of the suggested model are described subsequently.

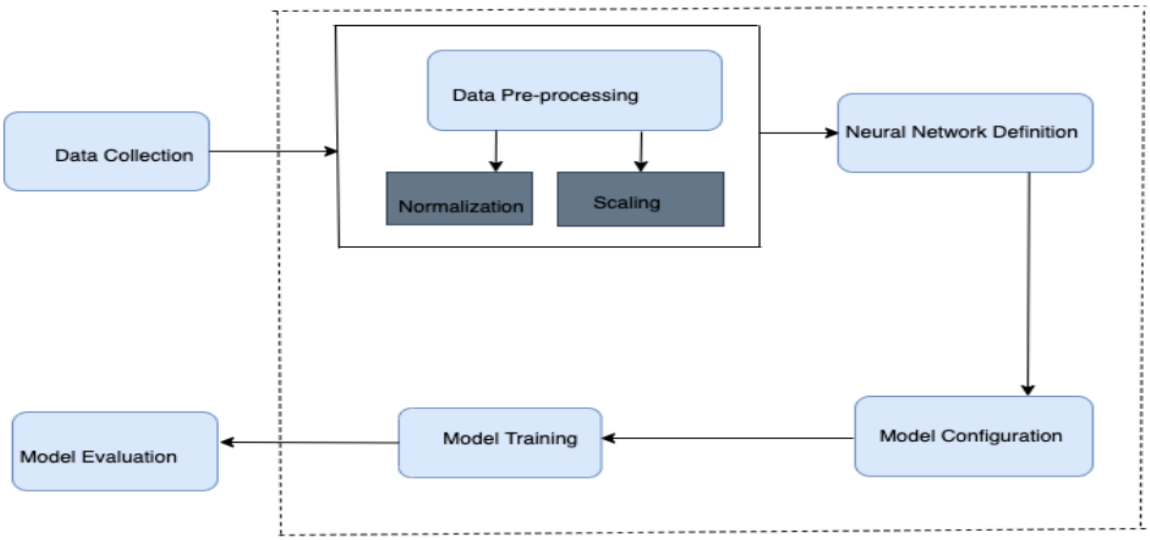

Figure 2. Overview of the model construction process

Dataset: The dataset for training and testing contains 7062606 instances of genuine traffic data from nine commercial internet-of-things devices that were attacked by the Bashlite and Mirai botnet attacks. This is a publicly available dataset obtainable at Kaggle.com and UCI machine learning (Dua & Graff, 2017) repositories, and is been used in previous works such as Meidan et al., 2018 and Mirsky et al (Mirsky et al., 2018). The dataset is divided into ten classes of attacks carried out by the two mentioned botnets and one normal traffic. This makes the data suitable for representation in a multi-class classification problem.

For each of the nine devices, we trained and optimized two deep learning-based classification models in addition to our proposed model. 80% of dataset was used for training and 20% for testing. Out of the 80% training set, we further split into 85:15 percentage ratio, where 15% is used for validation and the remaining 85% forwarded for training the models. The NBaIoT dataset is a bunch of data that contains both benign and botnet infected IoT traffic. The goal of this study is to write python code using deep learning to train a model that can classify the traffic data into one of three categories: namely benign traffic, mirai based traffic and bashlite based traffic. At the end, if a given IoT traffic is supplied to the model, it will tell us with certain confidence the class of that traffic. The one-versus-all classification method was extended to perform multiclass classification of the dataset using our proposed neural network.

Pre-processing the dataset: Relevant features were selected and standardized. Using the Keras library, the dataset was normalized too categorical. This is a multiclass classification problem so we needed to do one-hot encoding. Thus, all input data into the network needed to be converted into one hot encoding because we used crosse entropy as a loss function during implementation. This data is then supplied to fully connected neural network as shown in figure 1. We further trained two existing models: namely feed forward neural network and convolutional neural network-long short-term memory network on the same preprocessed data under the same platform. A comparison of the proposed model with the two existing model was performed to determine the efficiency and accuracy of the model vis a vis the existing models.

Various steps of code implementation of the proposed model; including pre-processing of the dataset, neural network model selection and model training and testing were done with python using the Keras library together with TensorFlow as backend. Auxiliary libraries such as Scikit learn, pandas, Seaborn and NumPy were imported and used.

Table 1: Sample of important features for the training of deep learning models that were taken from the NBaIoT dataset

Feature	Description
Network Host IP	statistics summary of current traffic from mean of outgoing traffic.
Network MAC IP	statistics summary variance of outgoing traffic
Network Channel	statistics summery of average incoming and outgoing traffic
Network Jitter	statistics summary of traffic deviation
Socket	statistics summary of average and variance of outgoing traffic jitter

Training our Neural Network

Training a deep neural network involves a process where we try to find the best set of weights to map the inputs to outputs in our datasets**.** For that, we chose categorical cross entropy as function to evaluate the set of weights. Stochastic gradient descent optimizer was then used to search through different weights for our network. This optimizer greatly influenced the performance of our model. After specifying the loss function and our optimizer, we collected and reported accuracy during each training iteration (epoch).

We run 500 epochs through the dataset, while splitting each epoch into a relatively small batch-size of 10. Through an exploratory process of trial and error, we selected these configurations with the aim to train the model so that it learns a good enough mapping of out input dataset to the output classification. Summary of the hyperparameter specifications of the network is shown in table 2.

Table 2: Summary of neural network configuration and hyperparameter specification.

Summary of 1D neuron construction
Model Input=N-BaIoT dataset, Feature dimension = 115
Convolutional layers = 5, size of filters=32, number of filters used =32
Maximum Pooling layer: Size =2,4,8,16, number of filters =32
Conv1D kernel size=5, strides=1 padding="same"
Dense (hidden) layer=2 units, activation function=ReLU
Output layer = 3 units, activation function=SoftMax

Evaluation Metrics of the model

Using python's metrics library, sklearn, Accuracy (A), Recall(R), Precision (Prec) and F-1 Score(F1) were tracked and recorded during the training epochs to assess the effectiveness of the suggested model as well as the existing models. The detailed definitions of the metrics are presented in table 3 below.

Table 3: Evaluation metrics.

Evaluation	Formula	Explanation
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$	TN = instances correctly classified as benign traffic. FN = instances incorrectly classified attack as benign traffic.
Precision	$\frac{TP}{TP+FP}$	incorrectly classified benign instances as an attack
Recall	$\frac{TP}{TP+FP}$	F1 score is the harmonic mean of precision and recall
F1-score	$\frac{(2\times \mathrm{Prec}\times R)}{\mathrm{Prec}+R}$	F1-score is the harmonic mean of precision and recall

Experimental Setup: Environment

The simulation experiments were performed on a physical machine running on 1.4 GHz Quad-Core Intel Core i5 with 8 GB 2133 MHz LPDDR3 memory and macOS Monterey,64 bits. An NVIDIA K80 GPU with 64 GB memory sourced from Google Collaboratory is used as an accelerator.

RESULTS AND DISCUSSIONS

The NBaIoT dataset provided for botnet attack research was provided by nine internet of things devices (Meidan et al., 2018). For this study, total of 5,000,000 data items put into separately grouped files with each file containing 115 features and output class label. The dataset has been designed with output class labels having the values; benign traffic, mirai based attack traffic and bashlite based attack traffic to give us a multiclass classification label. In all,555932 instances of the IoT traffic are normal traffic, representing 7.23%, while 7134943 instances are attack traffic.

Further distribution of the dataset sets 36.9% as bashlite based attack instances and 55.9% representing mirai botnet attacks.

Performance evaluation of deep learning classifiers

We present the result of our model in comparison to existing learning models trained with the same dataset under the same environment

Table 4: Comparison of our proposed model results with results from existing learning models. Existing models were trained on the same preprocessed data

Learning model	Accuracy	Precision	Recall	F1 - score
Feed forward neural network (ANN)	0.872	0.728	0.874	0.795
Logistic Regression Model	0.985	0.982	0.965	0.973
Our model	**0.987**	**0.980**	**0.973**	**0.976**

Comparison with existing studies

Table 5: Comparison of the performance of our study other existing studies. The existing studies also used the NBaIoT

Study in reference	Accuracy	Recall	Precision	F1 -score
(Mcdermott et al. McDermott et al., 2018)	99.0	98.0	98.0	-
(Apostol et al. Apostol et al., 2021)	99.7	99.0	99.0	-
(Rahmantyo et. al. Rahmantyo et al., 2021)	88.67	88.67	88.53	-
(Alkahtani et al. Alkahtani & Aldhyani, 2021)	87.19	89.23	87.76	89.64
(Hezam et al. Hezam et al., 2021)	89.75	-	-	-
Our study	**98.9**	**97.80**	**98.30**	**98.10**

DISCUSSION

The N-BaIoT dataset used for this study allowed for empirical analysis of IoT traffic using real traffic data collected from physical devices. This is significant because most of the prior experimental investigations on the identification of botnet attacks in IoT used simulated datasets. However, botmasters continue to design batch files of the bots which simulated dataset cannot capture and therefore results in poor detection models. Our study has an implication to generating new insights in support of the growing search for alternative approaches such as deep learning in detecting current sophisticated botnet activity that has surfaced in the internet of things landscape.

By using convolutional layers to extract various strange and stealthy patterns developed by the botnets, we employed an extension of "one-versus-all" method to do multi-class classification of IoT traffic into "Normal traffic"," Mirai based attacks" and "Bashlite based attacks".

Once the models were created, we trained other two already existing models on our training dataset and the performance results recorded. The training procedure was influenced by key hyperparameters such as number of training epochs, number filters, length of filters and number of hidden layers in configuring our proposed network summarized in table 2.

We present a comparison of the results of our model with results from existing models proposed by other scholars. We achieved the results presented in table 5 by training these models on same training dataset using the same hardware specifications used to train our proposed model.

CONCLUSIONS

This paper has presented a 1D deep neuron method to undertake multiclass classification of IoT traffic. Using experimental investigations, the model was evaluated using 15% of the entire dataset. Our model achieved accuracy score of 98.9%, with 98.3% precision. Recall was 97.8% and F 1 score 98.1%. These performance figures are quite high and gives an indication that our model can determine any unknown traffic passing through an IoT device as normal traffic or botnet attack traffic. If it is a botnet traffic, the model can tell us whether it is one of mirai based attack, bashlite based attack or something else, which would call for further investigations.

Future investigations may want to apply this convolutional neural network based deep learning approach to botnet attack analysis in other domains such as smart grid. Consideration should be given in selecting major parameters such as length of filters, filter numbers and number of hidden layers that can influence the training and ultimate performance of the neural network in those domains.

REFERENCES

Abdul Kadir, A. F., Stakhanova, N., & Ghorbani, A. A. (2015). Android botnets: What urls are telling us. *International Conference on Network and System Security*, 78–91.

Alkahtani, H., & Aldhyani, T. H. H. (2021). Botnet attack detection by using CNN-LSTM model for Internet of Things applications. *Security and Communication Networks*, *2021*.

Alqatawna, J., Faris, H., & others. (2017). Toward a detection framework for android botnet. *2017 International Conference on New Trends in Computing Sciences (ICTCS)*, 197–202.

Amini, P., Araghizadeh, M. A., & Azmi, R. (2015). A survey on Botnet: Classification, detection and defense. *2015 International Electronics Symposium (IES)*, 233–238.

Anwar, S., Zain, J. M., Inayat, Z., Haq, R. U., Karim, A., & Jabir, A. N. (2016). A static approach towards mobile botnet detection. *2016 3rd International Conference on Electronic Design (ICED)*, 563–567.

Apostol, I., Preda, M., Nila, C., & Bica, I. (2021). IoT botnet anomaly detection using unsupervised deep learning. *Electronics*, *10*(16), 1876.

Azmoodeh, A., Dehghantanha, A., & Choo, K.-K. R. (2018). Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning. *IEEE Transactions on Sustainable Computing*, *4*(1), 88–95.

Doshi, R., Apthorpe, N., & Feamster, N. (2018). Machine learning ddos detection for consumer internet of things devices. *2018 IEEE Security and Privacy Workshops (SPW)*, 29–35.

Dua, D., & Graff, C. (2017). UCI Machine Learning Repository. University of California, Irvine. *School of Information and Computer Sciences*.

HaddadPajouh, H., Dehghantanha, A., Khayami, R., & Choo, K.-K. R. (2018). A deep recurrent neural network based approach for internet of things malware threat hunting. *Future Generation Computer Systems*, *85*, 88–96.

Hezam, A. A., Mostafa, S. A., Ramli, A. A., Mahdin, H., & Khalaf, B. A. (2021). Deep Learning Approach for Detecting Botnet Attacks in IoT Environment of Multiple and Heterogeneous Sensors. *International Conference on Advances in Cyber Security*, 317–328.

Hoang, X. D., & Nguyen, Q. C. (2018). Botnet detection based on machine learning techniques using DNS query data. *Future Internet*, *10*(5), 43.

Jung, W., Zhao, H., Sun, M., & Zhou, G. (2020). IoT botnet detection via power consumption modeling. *Smart Health*, *15*, 100103.

LeCun, Y., Bengio, Y., Hinton, G., & others. (2015). Deep learning. nature, 521 (7553), 436-444. *Google Scholar Google Scholar Cross Ref Cross Ref*.

Letteri, I., Del Rosso, M., Caianiello, P., & Cassioli, D. (2018). Performance of Botnet Detection by Neural Networks in Software-Defined Networks. *ITASEC*.

McDermott, C. D., Majdani, F., & Petrovski, A. V. (2018). Botnet detection in the internet of things using deep learning approaches. *2018 International Joint Conference on Neural Networks (IJCNN)*, 1–8.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D., & Elovici, Y. (2018). N-baiot—network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing*, *17*(3), 12–22.

Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: an ensemble of autoencoders for online network intrusion detection. *ArXiv Preprint ArXiv:1802.09089*.

Nguyen, H.-T., Ngo, Q.-D., & Le, V.-H. (2020). A novel graph-based approach for IoT botnet detection. *International Journal of Information Security*, *19*(5), 567–577.

Pieterse, H., & Olivier, M. S. (2012). Android botnets on the rise: Trends and characteristics. *2012 Information Security for South Africa*, 1–5.

Rahmantyo, D. T., Erfianto, B., & Satrya, G. B. (2021). Deep residual cnn for preventing botnet attacks on the internet of things. *2021 4th International Conference of Computer and Informatics Engineering (IC2IE)*, 462–466.

Ramachandran, A., Feamster, N., Dagon, D., & others. (2006). Revealing botnet membership using dnsbl counter-intelligence. *Sruti*, *6*, 49–54.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Yerima, S. Y., & Alzaylaee, M. K. (2020). Mobile botnet detection: a deep learning approach using convolutional neural networks. *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 1–8.