Deep Learning Techniques to Detect Botnet

Doan Trung Son, Nguyen Thi Khanh Tram, Pham Minh Hieu

Abstract— **Over the past time, the world has witnessed an unprecedented explosion of Deep Learning. Besides the development of Information Technology, security and safety threats are also increasing, one of which is the Botnet network. Botnet network is increasingly complex and difficult to detect, and traditional techniques are no longer effective, so one of the urgent problems today is to find an effective solution to detecting botnets [2]. Based on the characteristics of deep learning such as scalability, performance, execution time, interpretability, etc., therefore, in this paper, the author proposes to use deep learning techniques to detect Botnet networks.**

Tóm tắt— **Thời gian qua, thế giới chứng kiến sự bùng nổ một cách mạnh mẽ chưa từng có của Deep Learning. Bên cạnh sự phát triển của Công nghệ thông tin, các mối đe doạ về an ninh, an toàn cũng ngày càng tăng lên, một trong những mối đe doạ đó chính là mạng Botnet. Mạng Botnet ngày càng phức tạp và khó phát hiện, các kỹ thuật truyền thống không còn phát huy được nhiều tác dụng, vì vậy một trong những vấn đề cấp thiết hiện nay đó là tìm ra được một giải pháp thật hiệu quả trong phát hiện mạng Botnet [2]. Dựa trên những đặc điểm của học sâu như: khả năng mở rộng hiệu suất, thời gian thực hiện, khả năng diễn giải… do đó, trong bài báo này, tác giả đề xuất sử dụng kỹ thuật học sâu để phát hiện mạng Botnet.**

Keywords—botnet; deep learning; BoTShark-SA; BoTShark-CNN.

Từ khóa— botnet; học sâu; BoTShark-SA; BoTShark-CNN.

I. INTRODUCTION

A BotNet is the shorthand term for "Bots Network". Just a network of infected computers (Bots/Zombies) dominated by another computer. The larger the Botnet network, the higher the danger. Botnets are actually a group of compromised Internet devices that are controlled remotely by cybercriminals. Cybercriminals use Botnets to launch coordinated attacks and perform other malicious activities. The word “botnet” is a combination of two words, “robot” and “network”. Here, a cybercriminal performs the role of a “botmaster” using a Trojan virus to compromise the security of several computers and connect them to the network for malicious purposes. Each computer on the network acts as a "bot" and is controlled by the bad guys to spread malware, spam, or malicious content to launch the attack. The number, scale, level of danger, and especially the hidden ability of Botnet networks is increasingly sophisticated and complex. In Vietnam, according to information from the Vietnam Computer Emergency Response Center (VNCERT), in the first two quarters of 2019, nearly 100,000 Vietnamese network (IP) addresses were queuing and connecting to Internet sites every day. computer network (Botnet) and up to 6,219 incidents of cyberattacks on Vietnamese websites.

BotHunter is one of the earliest behavior- based Botnet detection systems, it works based on the use of SNORT software to generate alerts about the behavior of each individual machine. However, the fact that it works by observing packet payloads makes this system not very effective in detecting Botnet networks that have encrypted connections. So BotMiner appeared, BotMiner works by grouping the behavior of different machines in the same Botnet.

On the other hand, in this early time, there was some research on how to detect malicious network flows by Botnet based on machine learning, but these network flows are only from internal machines in the LAN. After that, machine learning became a popular technique in Botnet detection. The two-stage system includes a feature extraction phase and a machine learning phase. This system is used to detect Botnet networks based on IRC (Internet Relay Chat - a protocol designed for real-time chat communication based on Client-Server architecture), it works by using the Bayesian classification method to detect network traffic from Botnet's C&C (Command-and-Control) machine, this system results in 90% detection rate and 15.4%

rate false detection rate (false detection rate is still high) [4].

Several other Botnet detections work based on DNS queries, and it has achieved a malicious DNS query detection rate of up to 92.5% [3].

However, detection based on DNS queries makes the systems workable only with Botnets that use the DNS system to search their C&C servers (mostly these networks are Botnets). concentrate).

The system for detecting P2P Botnet network flows works by assuming that the network traffic generated by users will fluctuate wildly unlike the network traffic of P2P Botnet. This system achieves detection rates up to 98%, but false detection rates are high (30%) [4].

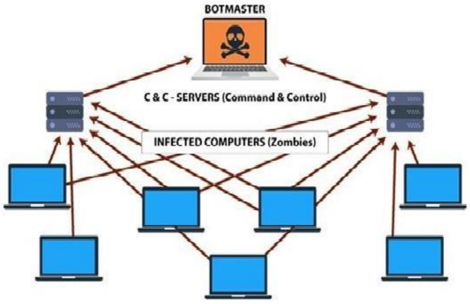

BOTMASTER
C & C - SERVERS (Command & Control)
INFECTED COMPUTERS (Zombies)

Figure 1. Typical structure of a Botnet

Botnets spread very quickly along with that, Botnet types are increasingly evolving and developing, which makes Botnet types increasingly dangerous and difficult to detect. Due to its ability to spread easily, any industry, profession, or field is at risk of being compromised by Botnet. So that when a Botnet signal appears, it is possible to investigate and detect if a device has been infected, how the infection took place, and how over a period of time the attacker has performed the following activities. what's on the device. Today, the network environment is becoming more and more complex, so security requirements are more difficult than ever. And with the Botnet being controlled by the botmaster through the C&C channel, in addition to being easy to hide, it also makes the Botnet an effective tool for cybercriminals to perform various types of malicious behavior. Some of the behaviors that Botnets can perform are: spamming, phishing, click fraud, distributed denial of service (DDoS) attacks, and malicious program distribution... Botnets are commonly used. frequently in Distributed Denial of Service (DDoS) attacks. An attacker can control a large number of hijacked computers at a remote station, exploit their bandwidth and send connection requests to the target machine. Many networks suffer terrible consequences after suffering attacks of this type [5].

II. BOTNET DETECTION AND DEEP LEARNING TECHNIQUES

A. Botnet detection techniques

Haddidadi and Zincir-Heywood [8] selected the feature to detect anomalies. The two-stage early Botnet detection method was implemented by Wang and Paschalidis [9]. Stevanovic and Pedersen [8] introduced a data stream-based Botnet detection method. The random forest algorithm achieves 94% accuracy. Kirubavathi and Anitha [8] introduced a Botnet-based detection method that modeled the behavior of network traffic using data flow characteristics and supervised machine learning (ML) methods. Nogueira et al. [7] proposed a method of Botnet detection based on characteristic network traffic patterns. Guntuku et al. [8] integrated a regular Bayesian model to preprocess the features and select the most representative set of features. Although the size of the benign network traffic dataset, and their categories are not given, the proposed method still detects the Botnet with 99.2% accuracy.

The author has evaluated 3 methods of Botnet detection based on deep learning: BoTShark-SA (Botnet Traffic Shark - Stacked Autoencoders); BoTShark-CNN (Botnet Traffic Shark - Convolutional Neural Networks) and Botnet detection method based on anomalous data flow. The results show that BoTShark-SA and BoTShark-CNN both achieve the ratio of correctly predicted positivity to true positive rate TPR (True Positive Rate) greater than or equal to

0.91 on the ISCX dataset. . BoTShark works on all basic properties and none of the properties are filtered by experts. BoTShark-SA achieved a TPR of 0.91 and a False Positive Rate (FPR) of 0.15, while BoTShark-CNN achieved TPR higher than 0.92. The comparison shows that the proposed deep learning-based approach achieves higher Botnet detection accuracy with a very low false-positive rate.

The results of the evaluation of the three methods above show that these methods are superior to other related methods, and it can detect Botnets with an accuracy of up to 99%.

Along with that, the model training time is acceptable when applied to a fairly large set of Botnets, the detection results are very accurate and have great potential for practical application. Besides, the Botnet detection model built based on the Artificial Neural Network (ANN) with the output of the Softmax output activation function is also the most applied and most effective model.

B. Deep learning technique to detect Botnet

In this paper, the authors build a classification model based on deep learning, in which: The model is built based on the activation function which is the Softmax function, the model is trained and tested with train and test sets. 10 best features have been extracted previously. The model is built on top of Keras with TensorFlow support. The application used is the Google Collaboration application by Google.

Introduction of dataset: The data set used by the author is a Bot-IoT built by Nickolaos Koroniotis, Nour Moustafaa, Elena Sitnikova, Benjamin Turnbull [6] from the University of New South Wales Canberra, Austria. The dataset is a combination of often simulated IoT network flows, along with diverse attack methods. The dataset is huge with 72000000 records, with 16.7 GB for CSV files and 69.3 GB for Pcap files.

The Bot-IOT dataset is a data set that has been built and analyzed thoroughly and completely, through which machine learning and deep learning methods can be easily applied. This dataset has shown its advantages when compared with other datasets as follows:

TABLE I. Bot-IOT DataSET and other dataset

Data set	Experiment	Actual traffic	Labeled data	IoT device traces	Diverse attack scenarios	Capture full packets	Extract new properties
Darpa98	T	F	T	F	T	T	F
KDD99	T	F	T	F	T	T	T
DEFCON-8	F	F	F	F	T	T	F
UNIBS	T	T	T	F	F	T	F
CAIDA	T	T	F	F	F	F	F
LBNL	F	T	F	F	T	F	F
UNSW-NB15	T	T	T	F	T	T	T
ISCX	T	T	T	F	T	T	T
CICIDS 2017	T	T	T	F	T	T	T
TUIDS	T	T	T	F	T	T	T
Bot-IoT	T	T	T	T	T	T	T

The dataset is built on three components, namely: Networking platform, simulation of IoT devices, and finally feature extraction and investigation, and analysis. First, the networking platform consists of normal and attack virtual machines (VMs). Second, about simulating IoT devices, IoT devices will be simulated through the Node-red tool. Finally, it is about extracting and investigating and analyzing features. Here, the Argus tool is used to extract features about the data so that machine learning techniques can then be applied.

Data preprocessing: First, it is necessary to import the dataset. First, we initialize the path to the dataset, and then use Python's CSV- formatted data reading function.

Next, we will get the data features, here the feature columns used to identify the data stream saddr, daddr, proto, sport, dport will be removed. In addition, we also remove the label columns in the data such as an attack, category, and subcategory. These are the labels of the data streams so we keep them separate.

In addition, the label used to distinguish it is the category. We need to convert the labels of the category column to numeric values. It is similar to a dictionary of 5 values: {0:A, 1:B, 2:C, 3:D, 4:E.} where label A is replaced with 0, label B is replaced with 1. Then we need to convert the numbers corresponding to the labels into vectors. For example, if the first column label is 0, it will convert to a 5-dimensional vector of [1,0,0,0,0], the second column label of 4 will convert it to [0,0,0,1,0]. Because the explosive lattice model requires the input of the label to be a vector, we need to convert its label to a vector.

For the extracted dataset, we proceed to classify the columns in the dataset into alphanumeric columns.

```
all_keys = X_train.keys()
num_keys = X_train.iloc[:, :-
2].select_dtypes(exclude=['object']).keys()
cat_keys = X_train.iloc[:, :-
2].select_dtypes(include=['object']).keys()
print(all_keys)
```

In which, the variable num_keys contains columns of numeric values and the variable cat_keys contains columns of literal values, the iloc command is used to browse each row in the column (keys) and the select_dtypes command is used to determine the data type to choose. Forcolumns with numeric values, we need to proceed to adjust the data of the characteristics to a common scale with a small enough range of values, the purpose of which is to help classifiers work properly. most accurate and effective. And the technique applied here is MinMax Scaling.

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

The above formula will normalize the properties to values in the range [0, 1]. However, because the values in the data set are too large, MinMax Scaling cannot be applied. Therefore, we first need to use the base 10 logarithmic formula to reduce the data size.

Similarly, for columns with literal values, we will also use MinMax Scaling to return data in the range [0,1]. However, before that, we also need to have a lexicographic method to convert alphanumeric values.

Building neural network model: The neural network model is built based on the output activation function, which is the Softmax function. The model consists of 3 layers as follows:

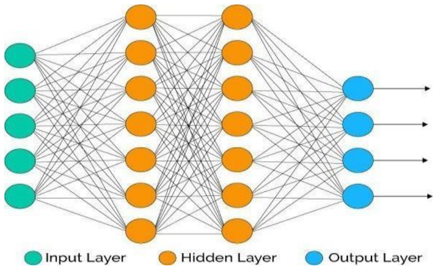


Figure 2. Neural Network model

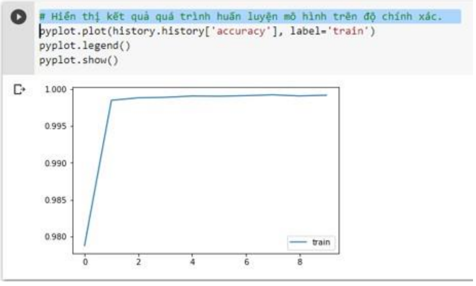
The input layer is a 14-valued attribute, followed by the first hidden layer with 128 nodes, then the second hidden layer with 64 nodes, and the last is an output label with 5 labels. Here, to increase model efficiency and reduce overfiting during training we use dropout technique.

After initializing the neural network, we proceed to train the model:

```
print("Bắt đầu huấn luyện mô hình Neural Network")
start = time()
history = model.fit(X_train, y_train_encode,validation_split
= 0.2, shuffle = True, batch_size=256, epochs=10, callbacks=[
stopping])
end = time()
print("Thời gian huấn luyện mô hình là: ", end - start)
Bắt đầu huấn luyện mô hình Neural Network
Train on 2347853 samples, validate on 586964 samples
Epoch 1/10
2347853/2347853 [==============================] - 51s
22us/step - loss: 0.5065 - accuracy: 0.9777 - val_loss:
0.1847 - val_accuracy: 0.9992
```

```
Epoch 10/10
2347853/2347853 [==============================] - 50s
21us/step - loss: 0.8242 - accuracy: 0.9989 - val_loss:
0.4445 - val_accuracy: 0.9999
Thời gian huấn luyện mô hình là:  499.8155708312988
```

Where epoch is the number of repetitions, and batch_size is the number of samples, these two values are calculated according to the given formula. After each iteration, we will drop out, edit the model to achieve the lowest loss and highest accuracy, and use the graph to display the training results based on the accuracy.



```
# Hiển thị kết quả quá trình huấn luyện mô hình trên độ chính xác.
pyplot.plot(history.history['accuracy'], label='train')
pyplot.legend()
pyplot.show()
```

Figure 3. Show training results

III. RESULTS AND DISCUSSION

A. Model evaluation

To make the assessment simple and effective, the author has evaluated the model based on 4 common metrics: Precision, Recall, F1-Score and Accuracy, these measures have scores equal to the value average of each label. The evaluation will be done by running the algorithm 3 times and calculating the results, which are:

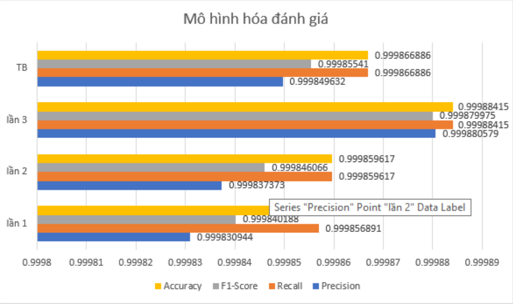


Figure 4. Modeling of column evaluation

Thus, we can see that the model has built accurate predictions with the ratio at 4 very similar measurements, and through 4 measurements we draw conclusions: With Precision 0.9997, it shows that every 10000 fields positive cases (Botnet) are predicted, then 9997 cases are true positives; Recall of 0.9996 shows that out of every 10,000 true positives, the model predicts 9996 of them. The remaining two measures, F1 and Accuracy, are both very high (0.9998 and 0.9996), also showing the effectiveness of the model in distinguishing between Botnet and normal.

To make the assessment more intuitive, the author calculated the measured value for each label: Precision, Recall and F1-score, and gave the results for each part as follows:

TABLE II. RESULTS FOR EACH PART

Precision	Recall	F1-score	Số mẫu	Precision
DDoS	1.00	1.00	1.00	385309
DoS	1.00	1.00	1.00	330112
Normal	0.85	0.58	0.69	107
Reconaissaine	1.00	1.00	1.00	18163
Theft	0.00	0.00	0.00	14

Figure 5 is a graph of modeling results by section with evaluation indicators: Precision, Recall, F1-score and the yellow line of the experimental sample index.

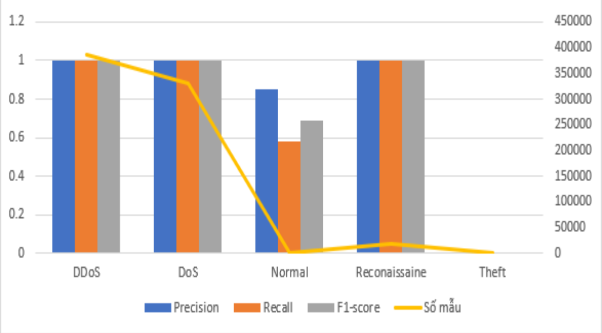


Figure 5. Modeling of partial results

The parameters of table 3 are determined as follows: TNR (Selectivity - Ratio of correctly predicted negative values to true negatives), NPV (Rate of correctly predicted negative values to value). predictive negative) and TS (Threat Score - just like F1-score would be the composite of Precision (PPV) and Recall (TPR)).

TABLE III. RESULTS OF THE REMAINING MEASUREMENTS

	TNR	NPV	TS	Sample
DDoS	0.999966	0.999871	0.999852	385309
DoS	0.999975	0.99997	0.999933	330112
Normal	0.999937	0.99999	0.535088	107
Reconaissaine	0.999943	0.999918	0.994512	18163
Theft	0.999981	1	0	14

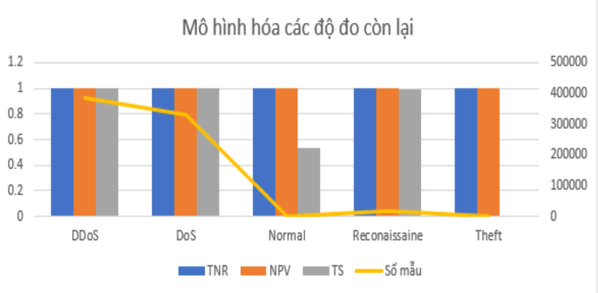


Figure 6. Modeling the remaining measures

Figure 6 models the measures: TNR, NPV, TS, and the yellow line of the experimental sample index.

From the measures for each component, we can draw the comment that when the number of positive samples of that type is large, the prediction accuracy of negative and positive values for that type will be more accurate. When the number of positive samples is small, for example, the Normal-type, the prediction will no longer be accurate, but because the difference between the number of positive and negative samples is too large, the measures of TNR and NPV are still give high results, similar to Theft's case. Therefore, the evaluation of the model's operability will be most accurate based on the following metrics: Precision, Recall, and F1- score.

B. Compare methods

In this paper, the author compares other machine learning methods including SVM (Support Vector Machine), RNN (Recurrent Neural Network), and LSTM (Long Short Term Memory) that have been introduced. in the paper "Towards the Development of Realistic Botnet Dataset in the Internet of Things for Network Forensic Analytics: Bot-IoT Dataset" by Nickolaos Koroniotis, Nour Moustafaa, Elena Sitnikova and Benjamin Turnbull [9] with DeepLeaning method.

TABLE IV. COMPARE METHODS

	SVM	**RNN**	**LSTM**	**DL**
Accuracy	**0.88372702**	**0.99740468**	**0.9974194**	**0.999611657**
Precision	**1**	**0.99990435**	**0.99991036**	**0.999722756**
Recall	**0.8837119**	**0.99749976**	**0.99750848**	**0.999631538**
Traning Time	**1270.48**	**8035**	**10482.19**	**499.815570831298 8**

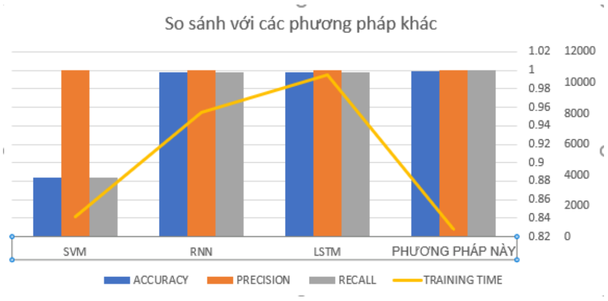


Figure 7. Compare methods

Thus, when compared with other methods, the applied deep learning method gives better results, along with a shorter training time. And also from the table, we can see that 2 other methods using neural networks, RNN and LSTM, also give high results compared to the SVM method. However, the Precision parameter of the SVM is 1, which shows that the SVM method predicts positive cases very accurately, not mistakenly predicting negative to positive, but it has the disadvantage that it is easy to miss a lot. positive case.

Therefore, it can be concluded that SVM method will be better than deep learning methods in predicting small data sets or data sets with few positive cases. However, within the scope of Network Forensics data, it is clear that the optimization of deep learning methods will be more clearly demonstrated.

IV. CONCLUSION

The article has drawn the role of deep learning in Botnet detection and realized Botnet network detection is based on deep learning. The article has outlined two methods of Botnet detection based on Deep Learning, which are BoTShark-SA and BoTShark-CNN based on 2 techniques and Autoencoder and CNN. The obtained results show the effectiveness and superiority of the Botnet detection deep learning model. In the future, the author wishes to directly execute on the raw data set after intercepting network data, thereby reducing data processing time and enhancing the applicability of the model. Building a deep learning application to detect Botnets can contribute to the task of ensuring information security, and protecting state secrets in the People's Public Security force in the coming time.

ACKNOWLEDGMENT

The authors would like to thank VINIF for their financial support to Nguyen Thi Khanh Tram as master student in VNU University of Engineering and Technology.

REFERENCES

[1] Đoàn Xuân Dũng, "Tóm tắt văn bản sử dụng các kỹ thuật trong Deep Learning", Luận văn ThS. Máy tính: 84801, 2018: 30-40.

[2] Hoàng Xuân Dậu, Nguyễn Trọng Hưng và Ninh Thị Thu Trang "Phát hiện botnet dựa trên học máy sử dụng dữ liệu truy vấn DNS: Phân tích ảnh hưởng của các đặc trưng huấn luyện", Hội thảo quốc gia lần thứ XXII Công nghệ thông tin và truyền thông. Thái Bình, 28-29/6/2019: 20-30; 100-120.

[3] Trần Thị Hằng, “Nghiên cứu tìm hiểu thực trạng về an ninh mạng và biện pháp khắc phục”. Đại học Dân lập Hải Phòng, 2016:30-40.

[4] Abu Rajab, M., Zarfoss, J., Monrose, F., & Terzis, A “A multifaceted approach to understanding the botnet phenomenon.” Hội thảo ACM SIGCOMM lần thứ 6 về Internet Measurement, 08/2006: 41- 52.

[5] Banday, M. Tariq, Jameel A. Qadri, and Nisar A. Shah, “Study of Botnets and their threats to Internet Security”, University of Kashmir, 06/05/2009: 9-24.

[6] Kronotis, Nickolaos “Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset”. Future Generation Computer Systems, University of New South Wales Canberra, Australia, 2019: 779-796.

[7] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton, “Deep Learning”, Nature, 2015: 436-444.

[8] Nogueira, António, Paulo Salvador, and Fábio Blessa, *"A botnet detection system based on neural networks*", Hội thảo quốc tế về điện tử truyền thông lần thứ 5, Piscataway, New Jersey, 2010: 57-62.

[9] Wang, Jing, and Ioannis Ch Paschalidis, "Botnet detection based on anomaly and community detection", IEEE Transactions on Control of Network Systems, 2016 4(2): 50-60, 2016.

ABOUT THE AUTHOR

Doan Trung Son

Workplace: Faculty of Information Security, People's Security Academy.

Email: son.doantrung@gmail.com

Education: - University: Faculty of Information Technology - People's Security Academy

Master: Faculty of Information Technology, Hanoi University of Science and Technology

Doctorate: Hagen University, Germany

Recent research direction: Cybersecurity, High-tech Crime Prevention, Artificial Intelligence, Data Science, Trust and Distributed Systems, Modern issues in information technology.

Nguyen Thi Khanh Trsm

Workplace: Phenikaa School

Email: *khanhtramt2k23@gmail.com*

Education: University: Faculty of Information Technology - People's Security Academy

Master: University of Engineering and Technology, Hanoi

Pham Minh Hieu

Workplace: Hatinh’s Police Email: *Hieu123cht@gmail.com*

Education: University: Technical Academy of Logistics Ministry of Public Security.