Computers & Security 113 (2022) 102549

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

On Detecting and Classifying DGA Botnets and their Families

Tong Anh Tuan[a,b], Hoang Viet Long[b,*], David Taniar[c]

[a] Vietnam Academy of Science and Technology, Graduate University of Science and Technology, Vietnam
[b] Faculty of Information Technology, University of Technology–Logistics of Public Security
[c] Faculty of Information Technology, Monash University, Australia

ARTICLE INFO

Article history:
Received 7 June 2021
Revised 3 October 2021
Accepted 12 November 2021
Available online 15 November 2021

Keywords:
Botnet detection
Dga botnets
Deep learning
Lstm network
Attention Layer
UMUDGA Dataset

ABSTRACT

Botnets are a frequent threat to information systems on the Internet, capable of launching denial-of-service attacks, spreading spam and malware on a large scale. Detecting and preventing botnets is very important in cybersecurity. Previous studies have suggested anomaly-based, signature-based, or HoneyNet-based botnet detection solutions. This paper presents new solutions for detecting and classifying families of Domain Generation Algorithm (DGA) botnets. Our solution can be applied in practice to disable botnets even if they have infected the computer. Our works help solve two problems, including binary classification and multiclass classification, specifically: (1) Determining whether a domain name is malicious or benign; (2) For malicious domains, identify their DGA botnet family. We proposed two deep learning models called LA_Bin07 and LA_Mul07 by combining the LSTM network and Attention layer. Our evaluation used the UMUDGA dataset recently published in 2020, with 50 DGA botnet families. The experimental results show that the LA_Bin07 and LA_Mul07 models solve the DGA botnets problem for binary and multiclass classification problems with very high accuracy.

1. Introduction

1.1. Botnets: an overview

Botnets are a concept that represents a network of computers that are illegally hijacked by hackers, with a scale that can range from a thousand to hundreds of thousands of computers in a particular botnet (Kim et al., 2011). They can conduct cyberattacks under the direction of a hacker, commonly known as a Bot Master, who manages these botnets.

The first Bot was GM, recorded in 1988–1989 by Greg Lindahl. They were developed based on the IRC protocol (Schiller et al., 2007). They are developed with good intentions to support system administrators. In 1999, Pretty Park discovered the first malicious code using IRC Server as a remote-control server, allowing the botnets to operate under the Client-Server model and opening the opportunity to develop the Bot network on a large scale. Currently, bots are constantly being developed with new methods of infiltration and stealth, posing challenges for security experts and system administrators.

Botnets can be deployed based on two models: Peer-to-Peer and Client-Server (Tuptuk and Hailes, 2018). A botnet usually consists of three components: BotMaster, Command & Control (C&C) server, and a network of bots (Ullah et al., 2013), illustrated in Fig. 1.

In this figure:

- BotMaster: A hacker who distributes bots and controls them through C&C server.
- C&C server: The servers that act as an intermediary between the BotMaster and the bots. They can administer bots, receive commands from the BotMaster to spread malicious code, forward commands to launch attacks, or distribute new updates.
- Bots: Computers or smart devices with an Internet connection. These devices are infected by malicious code. BotMaster manages them through the C&C server. They perform standard functions such as: hiding in the system, receiving commands, or receiving updates from the C&C server.

Botnets are usually distributed and hidden for most of their lifecycle and only perform attacks when they receive the command from hacker via C&C server. BotMaster can use botnets for the following purposes:

- Denial of Service Attacks: Botnets can launch a Distributed Denial of Service attack (DDOS) against any persons or servers on the Internet, using techniques such as SYN Flood, ICMP Flood, Tear Drop, Smurf Attack (Tuan et al., 2020). With their overwhelming advantage, they can damage any system. Some typical attacks include DDOS attacks on Amazon Web Services has

* Corresponding author.
E-mail addresses: longhv08@gmail.com (H.V. Long), David.Taniar@monash.edu (D. Taniar).

https://doi.org/10.1016/j.cose.2021.102549
0167-4048/

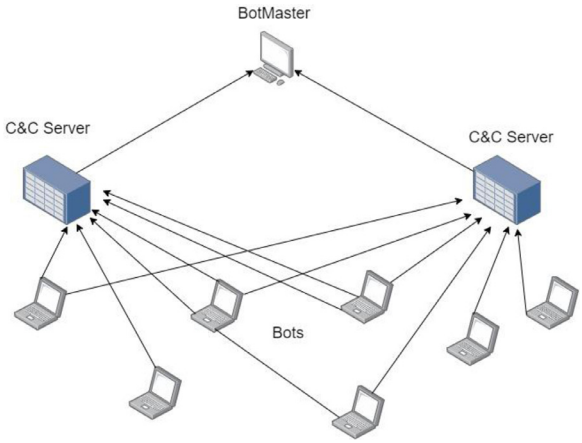


Fig. 1. Typical architecture of a botnet.

occupied bandwidth up to 2.3 Tbps (Nicholson, 2021); DDOS attack on OVH company with bots infecting IoT devices (E. U. A 2019).

- Send spam messages or e-mail: Botnets can send spam messages or e-mails, threatening users with less information security skills. They may accidentally open and perform the operations they require (Khan et al., 2015). Then, users are threatening of personal information stolen or malicious code installed on their computers. Although security tools can filter most spam messages or e-mails, their sending over the Internet also causes a significant waste of network traffic. By one estimate, about 85% of spam e-mails on the Internet are sent by botnets (John et al., 2009).
- Conduct information theft and espionage (Ianelli and Hackworth, 2007): Botnets can also use for intelligence and information gathering activities. They can secretly monitor user behavior, record keystrokes like a keylogger, or steal personal information, software copyright information stored on computers. This thread can pose a threat to all organizations and individuals, especially government agencies.
- Automated behaviors on the Internet: BotMaster can use botnets to impersonate ordinary users, conducting automated operations to benefit the BotMaster (Chen and Subramanian, 2018). It can be the behavior of clicking on ads to receive remuneration, or automatic comments, automatic votes on social networks. They cause discomfort for users as well as affect the revenue of companies in the field.

To sum up, the activities of botnets endanger the Internet; they bring a huge source of profits up to millions of dollars per month (Putman et al., 2018) for hackers. The European Network and Information Security Agency (ENISA) found that malware caused an estimated $13.2 billion in damage in 2006 to the world economy (Anderson et al., 2019). Content networks can also be leased to others by hackers to carry out illegal activities. The cost for hackers to maintain a 1000-machine botnet is about $7/hour and can be leased back for $25/hour (Li et al., 2021). Therefore, the detection and prevention of botnets have always been of interest to cybersecurity experts.

1.2. DGA botnets

The concept of Domain Generation Algorithm (DGA) botnets refers to the botnets deployed under the client-server architecture. The bots act as the client. After infecting the victim's computer, it will connect to the C&C server to receive commands. They usually look up the IP address of the C&C server through a DNS query. According to an automatic domain name generation algorithm, these domain names are constantly being changed to bypass security systems.

Looking at it in detail, let us review the main lifecycle activities of botnets. The life cycle of botnets includes three stages, namely:

- Infection: This is the first stage, where C&C server distributes malicious codes and malware through channels such as messages, e-mails, or malware. Computers infected with these malicious codes will become bots, contacted and managed by C&C server.
- Stealth: During this phase, bots will hide from security tools such as anti-virus software from the search activities. They also communicate with the BotMaster through the C&C server to receive software updates and send the information they collect.
- Attack: At a specific time, bots receive a command from C&C server to launch an attack. It could be a denial of the service attack campaign, spreading malicious code, malware. This activity is also the stage where bots are most easily detected. In some cases, BotMaster may also actively remove unnecessary bots from their botnets.

During the stages of the botnets lifecycle, communication with the C&C server plays an important role. Bots need to know the IP address of the C&C server to connect. Fixing an IP address for a C&C server is too revealing and can be easily blocked by security solutions such as firewalls, intrusion detection and prevention systems (IDS/IPS).

Bots use domain names as an efficient secret solution to query IP addresses to solve the above problem. Two simple techniques were used, such as domain name and multihoming (Schiller et al., 2007). Accordingly, the domain name technique uses several domain names to point to the same destination IP address, while the multihoming technique allows multiple domain names to point to the same destination IP address.

This solution is not practical because of its simplicity. The solution to finding IP address through the autogenerated domain name is proposed, called Domain Generation Algorithm. C&C servers and bots will jointly agree on an algorithm for automatic domain name generation. From time to time, the generated domains will point to an IP specified by the BotMaster. This solution is much improved than the previous one, allowing bots to query the server with a lower detection rate. At the same time, C&C server can also easily hide or change its IP address and still allow bots to find it.

The DGA botnet detection problem focuses on determining whether a domain name is malicious or benign, which can be extended to a multiclass classification problem to accurately determine the domain name family of each type of botnet. Timely detection of these queries can effectively stop botnet's activities, even if they have infected the computer.

1.3. Botnet detection approaches

Currently, there are two main approaches used to detect botnets (Karim et al., 2014), shown in Fig. 2

(1) Honeynet-based techniques (hacker trap network).

(2) Techniques based on Intrusion Detection System. This approach can include some techniques, such as:

- Anomaly-based botnet detection;
- Signature-based botnet detection;
- DNS-based botnet detection.

Approach (1) builds honeynets, called trap networks, to secretly collect botnets' information and analyze their characteristics and behavior. In general, honeynet-oriented botnet detection systems have the advantage of being easy to build and requiring few sys-

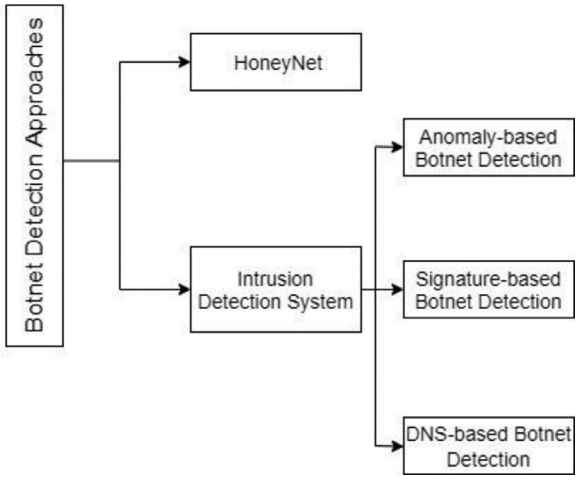


Fig. 2. Two approaches for botnet detection.

tem resources. However, systems of this type are often limited in scalability.

The second approach (2) is based on intrusion detection systems. IDS is a software application or a hardware device capable of monitoring system services to detect malicious behavior, security policy violations to notify management. IDSs are often installed with algorithms to monitor packets passing through network ports and then analyze to find botnets signs.

This study takes a botnet detection approach based on DNS analysis, detecting and classification malicious DNS connections. This approach has the advantage that it can prevent botnets from communicating with the C&C server. From there, it is possible to effectively disable botnets, even if they have somehow infected the computer. Besides, DNS analysis requires less computational resources than other solutions.

1.4. Our approach

Several solutions have been proposed to prevent botnets from infecting computers or communicating with the C&C servers. In this study, we approach a DNS-based botnet detection. We rely on deep learning to suggest new models capable of detecting and classifying botnets. It is possible to block connections from the bot to the C&C server and disable them. Thereby it is critical to disable the botnets even if they have infected the computers.

This paper proposes two new deep learning models to address binary classification and multiclass classification problems. Our approach is based on the LSTM network and Attention mechanism (Abakumov, 2016). For binary and multiclass classification, we proposed two new models, LA_Bin07 and LA_Mul07, respectively, to detect DGA botnets and classify their botnets families.

We conducted experiments using four datasets, including Abakumov's DGA Repository (Abakumov, 2016), OSINT DGA feed (OSINT 2021), UMUDGA Dataset (Zago et al., 2020), and 360NetLab Dataset (360NetLab 2016). The UMUDGA dataset that can be considered the most recent and complete was published in 2020, with 50 DGA families and a large number of domain samples. The assessment carried on this data set gives a more complete and objective view of the accuracy of the proposed model. Very few studies have been evaluated on this dataset.

Our results show that the LA_Bin07 model has high classification accuracy, reaching 98.32% on the UMUDGA set and up to 100.00% on the OSINT dataset. The LA_Mul07 model demonstrates an outstanding ability to classify DGA botnets families, much better than previous models, with accuracy reaching 99.72% and 85.88% on AADR, UMUDGA datasets, respectively. This evaluation shows that our proposed models can address the DGA botnets problems better than any other existing methods.

2. Related works

In this section, we summarize some techniques that have been proposed in previous studies, belonging to the DNS domain-based botnet detection approach.

2.1. Traditional techniques

In the study (Kwon et al., 2016), Kwon et al. argued that extracting important security information based on DNS traffic is possible. However, with the significant traffic of DNS queries, the analysis is complex. Studies are also negatively limited by many queries and domains, which requires a new approach. The authors presented a rapid and extensible approach called PsyBoG for malicious behavior detection based on extensive DNS traffic analysis. The proposed solution uses signal processing techniques and power spectral density (PSD) analysis to detect periodic DNS queries of the botnets. The PsyBoG solution has the following advantages: (1) Detecting botnets with sophisticated stealth capabilities; (2) Enabling large-scale processing of DNS queries; and (3) Being capable of detecting groups of servers with similar malicious behavior.

The proposed solution evaluated on 755 DNS traces collected in the real world, including malicious traffic, combined with 756 real DNS server traffic. Experimental results show that PsyBoG has an accurate detection rate of 95%, proving its effectiveness and applicability in practice when working with significant DNS traffic. The research team also detected 23 unknown botnets families in the above traffic, 26 known botnets families with a false positive rate of only 0.1%. However, the results of evaluations on other data sets of DGA botnets were not presented.

Wang et al. detected botnets based on domain analysis (Wang et al., 2017). They argued that botnets that use domain generation algorithms to hide are often difficult to detect by traditional techniques. They proposed a DGA-based botnet detection scheme called DBod. This solution is based on an analysis of the query behavior of DNS traffic, practical experiments on servers, and the fact that most malicious queries are intercepted. The feasibility of the solution is demonstrated by the network dataset obtained in the educational environment. This dataset was collected over 26 months from a network of approximately 10,000 users. The results show that the proposed solution is capable of detecting with an accuracy of over 99%. However, the limitation of the study is that the size and evaluation data set are minor, including only 04 families of DGA botnets considered including Kraken, Conficker, Cycbot, and Murofet, as well as not having been widely studied.

Chowdhury et al. commented that botnets are becoming more and more sophisticated and capable of causing more significant damage (Chowdhury et al., 2017). Most botnet detection methods based on rules (rule-based) or network traffic (flow-based) proved ineffective. Therefore, it is essential to develop a method with fast and robust detection. In their research, the authors propose a new botnet detection method based on the topology characteristics of the nodes in the graph, including degree, semi-out, semi-in, among out and semi-in, clustering coefficient. A clustering method based on a self-organizing map is applied to establish node clusters in the network based on the above-proposed properties. The proposed method can isolate bots in small-sized clusters while also containing most normal nodes in the same large cluster. The bots can then be discovered by looking for a limited number of makes. Besides, a filtering procedure is also proposed to improve the accuracy of the algorithm. The study was evaluated on the CTU-13 dataset (Yavanoglu and Aydos, 2017). The results show that the

proposed method can effectively detect bots, even though their behaviors are different. It should be noted that this method dramatically reduces the number of nodes that need to be searched, thereby providing faster discovery in large-scale networks.

In the study (Bisio et al., 2017), Bisio et al. presented a DGA botnet detection algorithm based on a single network monitoring. The stages of the proposed solution include: (1) Detecting a bot looking for a C&C server and related auto-generated domains; (2) Analysis of DNS requests that have been resolved within the same period. The linguistic and semantic features are then analyzed and classified into a particular DGA family. Finally (3), clusters are analyzed to minimize false positives. The proposed solution was evaluated on two environments, including a specially built network environment, with labeled DGA Domain families. The second network is a corporate LAN intranet. In the first experiment, all DGA domain families were detected. For the second experiment, the algorithm discovered an infected server during a 15-day evaluation process. The test data set includes 40 different DGA domain families, with the ability to detect most DGA domain families with an accuracy of 92.67% or more, except for one case reaching 88.85%. However, evaluation results on publicly available datasets have not been presented.

Wang et al. introduced an approach that includes: (1) botnets presence detection and (2) identification of infected nodes (Wang and Paschalidis, 2017). In the first stage, anomalies are detected by computing the significant difference between the current and previous states, called the empirical distribution. In the second stage, bots are discovered based on the idea of building a social network community in a graph, recording the correlation and interaction between nodes over time. Community detection is done by maximizing the module measure in this graph. They evaluated the proposed method using real-world botnets traffic and compared it with other studies.

Botnets are increasingly sophisticated and expand their scope of operation on many different platforms, stopping at computers and smart devices, IoT devices. In their research (Nguyen et al., 2020), Trung et al. showed botnets extension on IoT devices, and they have similar capabilities to traditional botnets types, completely capable of creating attacks. Large-scale denial of service. This threat increases when the security mechanisms for IoT devices are not adequate. IoT devices have their characteristics in terms of operating systems, processing capabilities, and power sources. Most of the previous studies have not solved multi-architecture and need to save processing resources on IoT devices. Requires a solution that requires fewer processing resources. However, the applicability of this technique on traditional families of botnets has not been detailed.

In their study, Trung et al. proposed a method of IoT botnet detection based on extracting attributes from PSI graph (PGS-Graph). Their solution can overcome the problem of multi-architecture in IoT devices while reducing computational complexity. The test results show that the proposed solution has an accuracy of 98.7%. The dataset consists of 11,200 ELF files containing 7199 IoT botnets samples and 4001 benign samples. It was collected from IoT-POT Team (Pa et al., 2016) and VirusShare ("VirusShare.com - Because Sharing is Caring." 2021). The comparison showed that the proposed model gives better results than with other studies. The source code of the solution is also published on GitHub.

Can et al. based on the essential characteristics of domain names as input to the algorithm to classify domain names as malicious or benign (Van Can et al., 2020). Specifically, they proposed extracting features such as domain name length, domain level, frequency of occurrence of characters, and creating a set of attributes. Next, the clustering algorithm on the neutrosophic set is applied to the binary classification problem. They evaluated their method on Alexa and 360NetLab datasets. The experiment result showed that the proposed method has higher accuracy than the compared algorithms. However, the applicability of the above algorithm to the multiclass classification problem has not been presented. Research has many potential new directions, but the accuracy achieved is not high enough to be applied in practice.

2.2. Machine learning-based techniques

Hieu et al. evaluated the effectiveness of supervised machine learning algorithms in detecting DGA botnets (Mac et al., 2017). With the input data of the domains, the research team built models including Hidden Markov, C4.5 Decision Tree, Extreme Learning Machine. At the same time, they also experimented on SVM, Recurrent SVM, CNN combined LSTM, and Bidirectional LSTM models. Evaluations were performed on a dataset of 1000,000 Alexa benign domains labeled non-DGA (Alexa Internet Inc. 2019) and 37 families of DGA domains labeled DGA. These DGA domain families are aggregated from the OSINT DGA feed from Bambenek Consulting, with 81,490 DGA Domains. The test results show that, in the binary classification problem, the SVM and LSTM-based models give results with an accuracy of 99.55% or more, significantly higher than traditional machine learning models. For the multiclass classification problem, the obtained results are not satisfactory, and especially there are eight families of DGA domains that supervised machine learning models do not detect. The author synthesized the experimental data set, so the ability to compare with other studies is limited.

In another study, Khan et al. looked at the discovery of Peer-to-Peer botnets (Khan et al., 2019). They argued that detecting botnets patterns of this kind poses more challenges than Internet Relay Chat IRC or the HTTP Hypertext Transfer Protocol. The research team proposed a multiclass traffic classification method by applying machine learning models to solve the above problem. First, the matching attributes will be selected, removing redundant attributes based on the decision tree algorithm. Non-P2P packets are also dropped to minimize traffic through network ports. Next, the second layer further classifies traffic into two groups, P2P and non-P2P. In the last layer, the research team proposed to use a decision tree classifier to detect P2P botnets. The average accuracy achieved was 98.7%. Experiments were performed on the CTU-13 and ISOT datasets (Yavanoglu and Aydos, 2017). However, the two datasets above do not contain as many DNS query patterns as the DGA botnets datasets.

Zago et al. (2020) presented a study on a new dataset for DGA botnet detection in a recent publication. They synthesized and built a new dataset called UMUDGA dataset. This dataset includes 1000,000 benign domains combined with 50 different DGA botnets families. Each DGA botnets family contains the source code for auto-generating domain names and sample domains from 10,000 to 500,000 for each domain family. The basic properties of the dataset were extracted and presented in the study. The advantage of the UMUDGA dataset is the aggregation of more domain names than previous data and publicity. Initial evaluations of the authors show that the SVM algorithm gives the highest F1-Score among traditional machine learning models, reaching 98.9% for the binary classification problem (Zago et al., 2020). For multiclass classification problems, these algorithms proved to be ineffective when the F1-Score was below 80%. UMUDGA dataset can be considered as the complete data set of DGA botnets today. However, there are currently very few published studies that have been evaluated on this dataset.

Botnets' network traffic always hides from regular user traffic, especially botnets networks deployed in the P2P model. In their study (Alauthman et al., 2020), Alauthman et al. proposed a mechanism to help reduce complex traffic, integrated with a reinforcement learning technique. The reduced network traffic helps to re-

duce the computational weight of the proposed algorithm. The test results show that the new method has a correct detection rate of 98.3%, a low false-positive rate of 0.012%. The study was evaluated on the synthesis of three datasets, including the IoT Dataset (Saad et al., 2011), the P2P botnets (Rahbarinia et al., 2013), and the Information Security Center of Excellence dataset (Shiravi et al., 2012).

2.3. Deep learning-based techniques

In an approach based on Deep Learning, Duc et al. (Tran et al., 2018) used Long Short-Term Memory Network (LSTM) to solve the DGA botnets problem. The research team solved two problems: binary classification and multiclass classification. The limitation of LSTM networks is multi-layer imbalance. From there, they proposed a new model called LSTM.MI, which combines both classification models, inherits the advantages of traditional LSTMs and reinforces them to improve accuracy and minimize noise maximumly. The dataset was collected from the real world, with 100,000 of the most popular benign domains from Alexa and 37 DGA domain families aggregated. The test results show that the proposed algorithm improves at least 7% accuracy than the original traditional LSTM model. It also achieves high accuracy in binary classification with an F1-score of 98.49% and can recognize five additional botnet families. The limitation of the study is that the data set is not complete, and some DGA domain families are almost undetectable.

Curtin et al. (2019) used RNN to detect and classify DGA botnets. They realized that domain names built on a sesame space had similar characteristics to benign domains, increasing the ability to hide the generated domain names. The team proposed a new concept called the Smashword scale. This is a scale that measures the similarity between a family of DGA domains and benign domains. The research team utilizes Recurrent Neural Network and Side Information network models to apply the above measurement criteria. The test dataset with 1000,000 Alexa domains, combined with 41 DGA botnet families aggregated by the research team, for a total of 2.3 million domains for both labels. Experiments show that the new model has potential applications by providing higher accuracy than previous models. Several families of complex DGA domains such as matsnu, suppobox, or rovnix were also discovered.

Simran et al. (2020) studied the problem of detecting malicious domain names generated by botnets or malicious e-mails and URLs. Several n-gram-based feature representation techniques have been used to model the problem. The dataset for evaluation includes benign and malicious domains collected from OpenDNS, Alexa and OSINT Feed. The comparison results show that the CNN-LSTM model is the most effective, with the F1-score of 96.3% for the binary classification problem.

Nugraha et al. (2020) published a complete study on the application of deep learning models to botnet detection. The models evaluated include CNN, LSTM, hybrid CNN-LSTM, and Perceptron (MLP). Evaluations were performed on the CTU-13 dataset. The model detected both known and unknown botnets families, showing the effectiveness of deep learning approaches on botnet detection. Variations of botnets are constantly appearing, so studies always need to be updated and evaluated on newer data sets.

2.4. Related works summary

We summarize some related studies in the following factors: methods, approaches, and evaluation dataset, presented in Table 1.

2.5. Proposed work

In detecting botnets based on DNS queries, previous studies have used techniques ranging from traditional to modern. Some techniques based on domain analysis (Kwon et al., 2016; Wang et al., 2017) or based on fuzzy set theory (Van Can et al., 2020) give quite a good accuracy. More modern techniques such as machine learning (Mac et al., 2017; Khan et al., 2019; Zago et al., 2020) or deep learning (Tran et al., 2018; Curtin et al., 2019; Simran et al., 2020; Nugraha et al., 2020) offer improvements and give higher results but require models more complex training. In general, the proposed techniques have good accuracy, but the above algorithms still need to be continuously improved for the following reasons:

- New variants of botnets are constantly being developed with better stealth algorithms. Therefore, the detection algorithms must also be constantly updated to keep up with such changes.
- Although some techniques were able to achieve high accuracy, however, for security reasons, we always need to raise the accuracy even higher to have the ability to protect the system at the highest level.
- Many previous studies were tested and evaluated on data sets collected by their author, limited public sharing, not favorable for comparison. At the same time, there are few studies evaluated on the newly published UMUDGA (360NetLab 2016) dataset in 2020.
- Some new studies only solve the problem of detecting botnets but have not detailed the problem of classifying the families of those botnets.

From those problems, in this study, we aim to solve the two most essential problems in botnet detection, including:

- Binary classification problem: With the input data of domain queries, this problem will classify domains into two classes, including benign and malicious domains.
- Multi classification problem: With the input data of maliciously labeled domain name queries, this problem will determine which domain name belongs to which DGA botnets family. That identification result can better assist in scanning and removing botnets from infected computers.

We developed two deep learning models, named LA_Bin07 and LA_Mul07, to solve the above two problems, respectively. Details are presented in Section 3 of this paper.

3. Proposed deep learning models

We have input data like domain names and output data as its corresponding label in detecting botnets based on DNS queries. The binary classification problem has two labels, 0 and 1, corresponding to benign domains or domains generated by botnets. The multiclass classification problem has n labels numbered from 1 to n, corresponding to the labels of the detected botnets family. An illustration of the labels to be classified for both problems is given in Table 2:

With our arguments and evaluation, we propose two new deep learning models, LA_Bin07 and LA_Mul07, to solve the above two problems.

3.1. LSTM network

Long-Short Term Memory Networks (LSTM) is developed from an RNN network, with the ability to learn dependencies further than RNN (Du and Swamy, 2014). LSTM was first proposed in 1996 by Hochreiter (1996) and has been continuously improved since.

Table 1
Summarize related works.

No.	Authors	Method	Datasets	Approaches	Results
1.	Kwon et al. (2016)	PsyBoG	DNS traces from real-world (49 DGA botnets families)	Behavior-based botnet detection	Accuracy: 95%
2.	Wang et al. (2017)	good	DNS traffic from an educational network for 26 months (04 DGA botnets families)	Domain analysis-based botnet detection	Accuracy: 99.6% (evaluation on 4 DGA bonets families)
3.	Chowdhury et al. (2017)		CTU-13 dataset (7 botnets families)	Graph-based botnet detection	Detecting bots by searching less than 0.1% of all nodes
4.	Bisio et al. (2017)		Actual data during 15 days (40 DGA botnets families)	Network monitoring	Anomaly indicator: 0.9806
5.	Wang and Paschalidis 92017)		Real-world botnets traffic	Anomaly-based botnet detection	Normal Mean: 0.0017; Bot Mean: 0.024; Difference: 0.0223
6.	Trung et al. (Nguyen et al., 2020)	PSI-Graph	Collected from IoTPOT Team and VirusShare	Graph-based botnet detection	Accuracy: 98.7%
7.	Van Can et al. (2020)	NCM	Alexa top 1.000.000 domains and 360NetLab datasets	Fuzzy clustering algorithm-based botnet detection	Accuracy: 81.09%
8.	Hieu et al. (Mac et al., 2017)	SVM and LSTM-based models	Alexa top 1.000.000 domains and 37 DGA botnets families	Machine learning and deep learning-based botnet detection	SVM: 88.85%; Bidirectional LSTM: 92.70%
9.	Khan et al. (2019)	P2P botnet detection	CTU-13 and ISOT dataset	Machine learning-based botnet detection	The average accuracy of 98.7%.
10.	Alauthman et al. (2020)		03 dataset, include: IoT Dataset, P2P botnets, and Information Security Center of Excellence dataset	Reinforcement learning-based botnet detection	The detection rate of 98.3%
11.	Zago et al. (2020)		UMUDGA dataset (50 DGA botnets families)	Machine learning-based botnet detection	- Binary Classification: F1-score reach 0.989 - Multi Classification: F1-score reach 0.769
12.	Duc et al. (Tran et al., 2018)	LSTM.MI	37 DGA botnets families collected	Long Short-Term Memory network	F1-score: 0.9298
13.	Curtin et al. (2019)	Split-glrt-lstm-aug	41 DGA domain families aggregated by the research team	Recurrent Neural Network and Side Information network	True positive rate of 95%
14.	Simran et al. (2020)	CNN-LSTM	Collected from OpenDNS, Alexa and OSINT Feeds	Convolution Neural Network and Long Short-Term Memory network	Accuracy: 94.0% F1-Score: 91.2%

Table 2
Illustrate the desired label of the binary and multiclass classification problem.

Example Domains	Binary Classification	Multi Classification	Describe
microsoft.com	0	None	Legit domain
howanyrepairyourconvey.ru	1	1	DGA Botnets, family 1: gozi_gpl
qkdchxxxzn.com	1	2	DGA Botnets, family 2: alureon
ofdhiydrrttpblp.com	1	3	DGA Botnets, family 3: cryptolocker

This algorithm works effectively on many different problems, especially string problems.

The structure of the LSTM network is also based on the chain architecture. However, each module in the chain is built with more improvement. Instead of having only one Neural network layer like RNN, LSTM has four layers in each module, and they can interact with each other. An illustration of the four layers of the LSTM is shown in Fig. 3. where

- The arrows indicate the direction of the vector in the model;
- Yellow rectangles: Representing a neural network layer, with activation function can be Sigmoid (Sibi et al., 2013) or tanh layer;
- Pink circles: Representing calculations between vectors, usually vector addition;
- The concatenate lines signify the merge, and the copy lines indicate its data are copied and moved to different places.

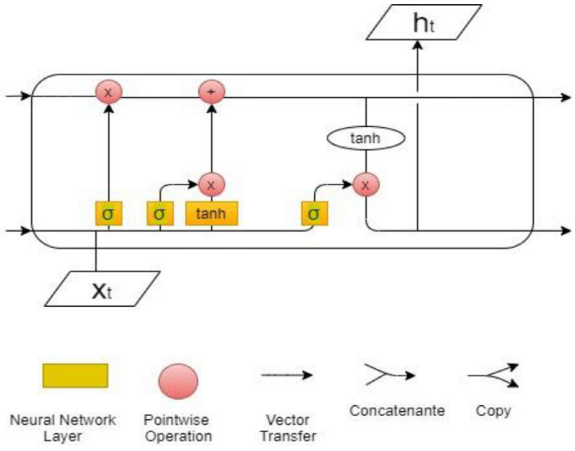


Fig. 3. The 4-tier architecture of the LSTM module.

3.2. Attention layer

Attention is a modern technique used in neural networks. This technique has proven effective in machine learning or natural language processing problems (Vaswani et al., 2017). Attention has the ability to find correlations between words in sentences and their roles in that sentence. We can identify what is the focus of attention in a sentence. Attention also allows the model to look at sen-

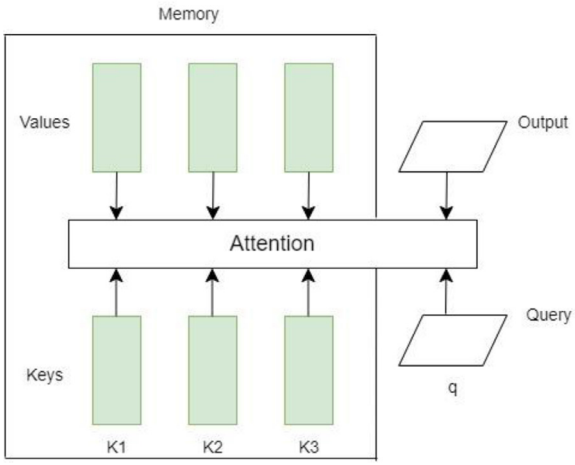


Fig. 4. The architecture of the attention layer.

tences in their entirety and match words with their contexts. This self-learning mechanism is called "self-attention". With the use of the attention mechanism, we can determine what is essential in the sentence or significantly influence the output label, thereby focusing on learning more to improve prediction accuracy.

Attention is a component consisting of 3 elements: Query, Key, and Value. They are proposed to help the training model focus more on a specific feature of the data, or in other words, more focused and essential information. The components of the attention layer are shown in Fig. 4:

Accordingly, query(q) will get the following information to be processed. They are divided into two parts, respectively, key and value. The i^{th} information is denoted by k_i and v_i.

For each q input, a_i is called the influence of the i^{th} information on q, calculated by

$$a_i = \alpha(q,\ k_i) \tag{1}$$

Then a_i are normalized to get b_i, the commonly used normalization function is softmax given in Gao and Pavel (2017),

$$b_i = \frac{\exp(a_i)}{\sum_j \exp(a_j)},\ b = [b_1,\ b_2,\ \ldots, b_n]^T \tag{2}$$

Finally, the v_i values are recalculated based on b_i. Normalization keeps the data from being resized compared to the model being trained. Attention mechanism helps increase attention to essential features of data, which can be extended to many other problems.

From the analysis of DGA botnets, we make the following comments:

- The input to the algorithm is DNS queries or domain names. If the domain is decomposed into words or letters, a domain name can be considered a sentence. Some words can represent the main body, which means that some keywords can represent a domain name feature.
- Domain names in the same family of DGA botnets have similar characteristics, built on the keyword sets of that domain family. Therefore, there is a difference in the specific keywords among the domain families that do not have the same family name.
- Domain names in the same family can create a familiar context for that domain name, based on the features of a set of keywords to generate that domain family. The contexts between different classes are different in the classification problem.

The attention mechanism has outstanding advantages that have been applied in the previous natural language processing studies, which is entirely feasible to apply to the problem of detecting DGA botnets. We propose a deep learning model that works according

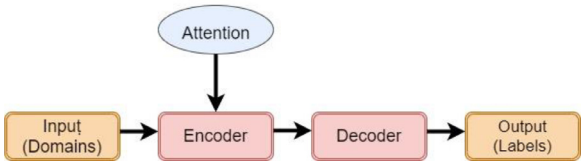


Fig. 5. Attention in our Sequential Model.

to the sequential model. The role of attention is shown in the encoder phase, illustrated in Fig. 5: whereas:

- Input: The dataset of domains;
- Encoder: The process of converting attributes to learn and train the model. This process involves attention.
- Decoder: The output of the encoder is the input of the decoder, which finds the probability distribution from the features learning in the encoder to determine the output label;
- Output: This is the label found through the model.

The encoder and decoder process is performed by model LA_Bin07 and LA_Mul07 proposed below.

3.3. Proposed model

The deep learning models proposed in previous studies are the basis for us to propose new models. The results (Mac et al., 2017; Tran et al., 2018; Simran et al., 2020; Nugraha et al., 2020) show that the LSTM model is suitable for the problem of detecting DGA botnets when achieving high accuracy. Some studies in natural language processing show that the attention mechanism (Vaswani et al., 2017) is suitable for remembering the essential features in a long sentence or a paragraph instead of memorizing the whole paragraph.

We observe that botnets' domains are always generated from a set of keywords, including predefined keywords and keywords that depend on current factors such as time, operating system. A domain name is automatically generated from that set of keywords. Among them, keywords play a more critical role than the rest, especially the predefined keywords specific to that family of DGA botnets. We also noticed a difference between the keyword sets of different DGA botnets families. It is essential to design the model to choose the right keywords.

The attention mechanism helps the model learn the important keywords located in a domain name, from which it can get the most vital features for the classification. This also reduces the computational load by the model. The reduction in time also makes sense when applied in practice.

From the above research results and arguments, this study proposes two new deep learning models, combining attention layer and LSTM network for binary classification and multiclass classification problems. The binary classification model (named LA_Bin07) is for classifying domain names as benign or malicious. Next, the multiclass classification model (named LA_Mul07) classifies the family of the malicious domain detected in the previous model.

Two models LA_Bin07 and LA_Mul07, are applied to the binary classification and multiclass classification problem, respectively, with the steps illustrated in Fig. 6.

First, the query domains will be fed into a trained binary classification model, labeling this as malicious or benign.

- For benign domains, the allowed access generally takes place in practice.

According to maliciously labeled domains, they are further passed through the multiclass classification to accurately identify their DGA botnets family.

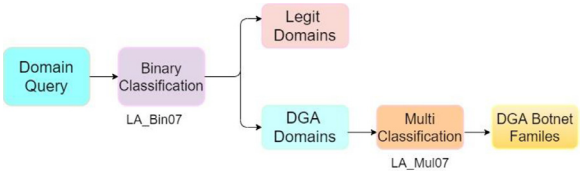


Fig. 6. Steps in the DGA botnet detection problem.

3.3.1. LA_Bin07 model for binary classification

With the domain query set as input, the LA_Bin07 model helps identify the benign domain class and the malicious domain name class.

In the binary classification model, besides the two main classes, LSTM and attention, we add the dropout layer to reduce the number of paragrams that need to be calculated and add a Dense layer to stretch the data back to the corresponding labels. The layout of the model's layers is shown in Fig. 7.

The model is designed above with two LSTM blocks incorporating the Attention layer, operating in a sequential model. First, data will be trained by the LSTM layer. The attention layer then selects the most critical features of the model. The Dropout layer (Hinton et al., 2014) is added after the Attention layer, which is used to reduce the parameters to be calculated, helping to reduce the training time of the model but still ensure to keep the essential features. While the Self Attention layer is attention's self-learning mechanism, the Seq Weighted Attention layer is weight-based attention. These two layers complement each other's advantages and disadvantages when combined with the LSTM network.

After the LSTM and Attention training process, the Dense layer (Vaswani et al., 2017) stretches the calculated results corresponding to the number of part classes to be classified. The Dense and Dropout layer continues to be repeated with two activation functions, respectively, Rectified Linear Unit (ReLU) and Sigmoid (Eckle and Schmidt-Hieber, 2019). The ReLU function simplifies the training model and reduces the computational load while retaining the required features. The sigmoid function is a linear product function, which is very suitable in the current binary classification problems. In addition, our review and evaluation results show that ReLU variants such as PReLU layer, Leaky ReLu, ELUs, and SoftReLU give no better results than the ReLU function. Details of the results are presented in the evaluation section.

The two activation functions used are ReLU and Sigmoid given by

$$y = \max(0,\ x) \tag{3}$$

$$\emptyset(z) = \frac{1}{1 + e^{-z}} \tag{4}$$

respectively.

The parameter x in the formula (3) (the ReLU activation function) and the parameter z in the formula (4) (the Sigmoid activation function) are the input values of the these activation functions, obtained from the two dense layers according to the proposed model in Fig. 7.

3.3.2. LA_Mul07 model for multi classification

For the Multi Classification model, some Dense and Dropout layers are removed, the activation function used is Softmax (Sharma et al., 2020), given by the following Formula 5, used to predict the label of the domain name

$$a_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)},\ \forall i = 1,\ 2,\ \ldots,\ C \tag{5}$$

where:

- C: The number of classes to be classified, numbered from 1 to C. In the multi-class classification problem, C is the number of DGA Botnet families to be classified;
- The set Z is the input value of the Softmax function, including the z_i with i from 1 to C. Z_i can be understood simply as the possibility that the data belongs to the i^{th} class. Thus, Z is the input value of the Softmax activation function.

The structure of the layers in the model is shown in Fig. 8.

For multiclass classification, the number of classes to classify corresponds to the number of families of DGA botnets in the dataset. The number of layers is quite a lot and can go up to 50 classes for UMUDGA datasets. We designed a simpler model than LA_Bin07, aiming to reduce the computational workload. However, the model is still guaranteed to be trained through two LSTM blocks combined with the attention layer. Self-Attention pays attention to features based on the self-attention mechanism, while Seq Weighted Attention is based on the learned weights. Two dropout layers on each block help reduce parameters. We have reduced a dropout layer compared to the LA_Bin07 model to avoid losing too much information, which may contain important information. This is especially significant when the number of classes to be classified is extensive, leading to the distinction between classes that can be blurred if the parameters to distinguish them are incomplete.

Next, only one dense layer is used, and the activation function is softmax. The softmax function allows activation at the final output layer of the model to give results suitable for the multiclass classification problem. They play an essential role in concluding the label of input data. Our studies also show that the softmax function is perfectly suitable for this problem.

4. Experiment and evaluation

4.1. Dataset and assessment environment

For the most objective and comprehensive evaluation results, we conducted our tests on 04 datasets, including Andrey Abakumov's DGA Repository (Abakumov, 2016), OSINT DGA feed (OSINT 2021), UMUDGA dataset (Zago et al., 2020), and 360NetLab dataset (360NetLab 2016), specifically as follows:

- Andrey Abakumov's DGA Repository (abbreviated as AADR): This repository was created in 2016 by Andrey Abakumov, including source code for automatic domain generation algorithms and malicious DGA domains. Alexa's top 1000,000 popular domains are also used and labeled as benign. This dataset is suitable for building a neural network to detect DGA botnets. Many previous studies have used it to evaluate their solutions. This dataset is publicly available on GitHub (Abakumov, 2016).

- OSINT DGA feed (abbreviated as OSINT): This is also a trusted dataset created by Bambenek, with a large volume of malicious domains collected and aggregated. This data is freely shared with the scientific community and is not permitted for commercial use (OSINT 2021).

- UMUDGA dataset (abbreviated as UMUDGA): Is a data set synthesized and built by a research team at the University of Murcia. This is the complete dataset available today when 50 DGA domain families are aggregated. The team also provided 10,000 to 500,000 samples of malicious domains for each of these DGA domain families. The data is arranged scientifically, in ARFF, CSV, and text formats, suitable for different tools and programming languages. All data is easily accessible on Mendeley Data (Zago et al., 2020), ensuring the reliability and public disclosure of the data.

- 360NetLab dataset (abbreviated as 360NetLab): This is a data set of malicious domains collected from the real world, built by the 360NetLab research team, Qihoo 360 Technology Co., Ltd

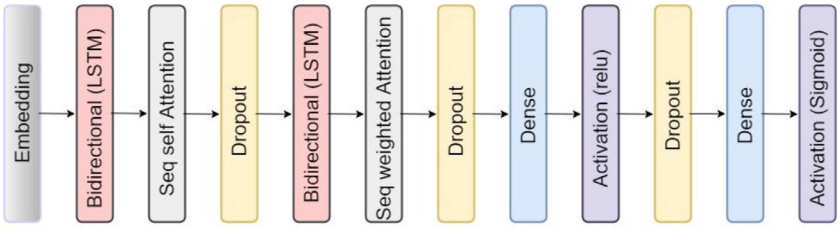


Fig. 7. Proposed structure of LA_Bin07 model.

Table 3
Description of the datasets used in the experiment.

Dataset	BinaryClassification	MultiClassification	Number of Legit domains	Number of DGA botnets domains	Number of DGA botnets families
APR	X	X	1.000.000	801.667	08
OSINT	X		1.000.000	495.186	
SMUDGE	X	X	1.000.000	500.000	50
360NetLab	X		1.000.000	1.513.524	

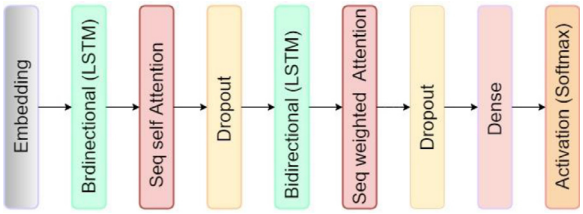


Fig. 8. Proposed structure of the LA_Mul07 model.

(360NetLab 2016). The search engine will continuously find and detect the latest DGA domain templates to update the database. However, the limitation of this dataset is the lack of stability when continuously changing in real-time.

We use the above 04 datasets for our evaluations because they fully meet the following properties:

- Originality: These datasets include benign and malicious domains in their original form, and the data has also been labeled, suitable for the input of research problems.
- Ease of comparison: Four data sets have been used by other researchers to evaluate in their previous studies. It is convenient for comparison between our results and previous results.
- Publicity: These datasets are publicly available on the Internet, easily accessible, and convenient for researchers.

In addition, some other datasets were not used for some reasons such as not containing original domains (Suryotrisongko, 2020), or being collected in real networks and limited publicity (Antonakakis et al., 2012; Zhou et al., 2013; Bilge et al., 2014; Nguyen et al., 2015; Sharifnya and Abadi, 2015; Wang et al., 2017).

We used all four datasets for the binary classification and two datasets, including Andrey Abakymov's DGA Repository and UMUDGA Dataset for the multi-classification. The datasets are split at 80%–20% for the training set and test set, respectively. The domain samples are randomly divided and the number of samples of each class also corresponds to the rate of 80%–20%, which ensures objectivity in the evaluations.

We use the training set to train the model separately. Then, the trained model will be independently evaluated on the testing set to ensure that the model will predict the unlearned data. We also conduct the evaluation using cross-validation with $k = 5$, then take the average value of the above evaluations to ensure that the obtained results most accurately reflect the model's performance.

More details of the number of domains used for the experiment are given in Table 3:

For the binary classification problem, with label 0 being legit domains, label 1 being DGA domains, we evaluated the Accuracy, Precision, Recall, and F_1-score, respectively calculated by Formula 6, Formula 7, Formula 8, Formula 9.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

$$F_1 - \text{score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$

where:

- TP: Number of samples of benign domain names that are properly classified as benign;
- TN: Number of samples of domains as malicious that are properly classified as malicious;
- FP: Number of malicious domain samples misclassified as benign;
- FN: Number of benign domain samples misclassified as malicious.

In the multiclass classification problem, we further evaluate the Micro average and Macro average of Precision and Recall. This helps the reader see the model's overall performance instead of just looking at each specific class. These values are calculated using the following formulas:

Calculate micro average:

$$\text{Micro average Precision} = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FP_c)} \tag{10}$$

$$\text{Micro average Recall} = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C} (TP_c + FN_c)} \tag{11}$$

- Calculate macro average:

$$\text{Macro average Precision} = \frac{\sum_{c=1}^{C} \text{Precision}_c}{C} \tag{12}$$

$$\text{Macro average Recall} = \frac{\sum_{c=1}^{C} \text{Recall}_c}{C} \tag{13}$$

where

	AADR	OSINT	UMUDGA	360NetLab
Accuracy	0.9906	1.0000	0.9832	0.9939
Precision	0.99	1.00	0.98	0.99
Recall	0.99	1.00	0.98	0.99
F1-Score	0.99	1.00	0.98	0.99

Fig. 9. Test results of LA_Bin07 model for binary classification.

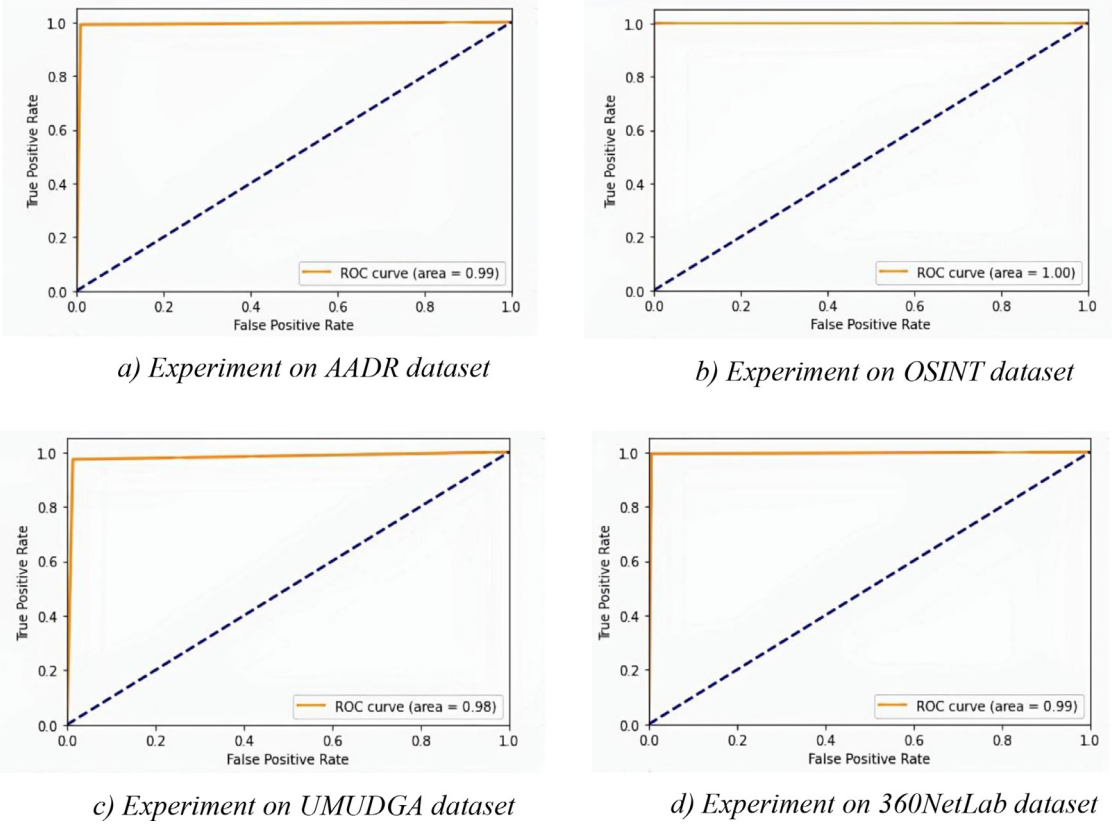


Fig. 10. The graph of ROC Curve and AUC plots (area) on experiments.

- C is the number of classes to be classified, c is the ordinal number of the class being considered.
- Based on Precision and Recall, Micro-average F1-Score and Macro-average F1-Score are calculated similarly to F1-score, respectively. In addition, we also conduct detailed model evaluation through ROC curve and AUC.

The model was trained and assessed on Google Colab Pro, Linux environment, using NVIDIA Tesla T4 GPU, Keras library.

4.2. Test results of the binary classification

For the binary classification problem, the test results through the parameters Accuracy, Precision, Recall, and F_1-Score are given in Fig. 9:

The corresponding ROC curve and AUC plots (area) for 04 datasets AADR, OSINT, UMUDGA, and 360NetLab are shown in Fig. 10.

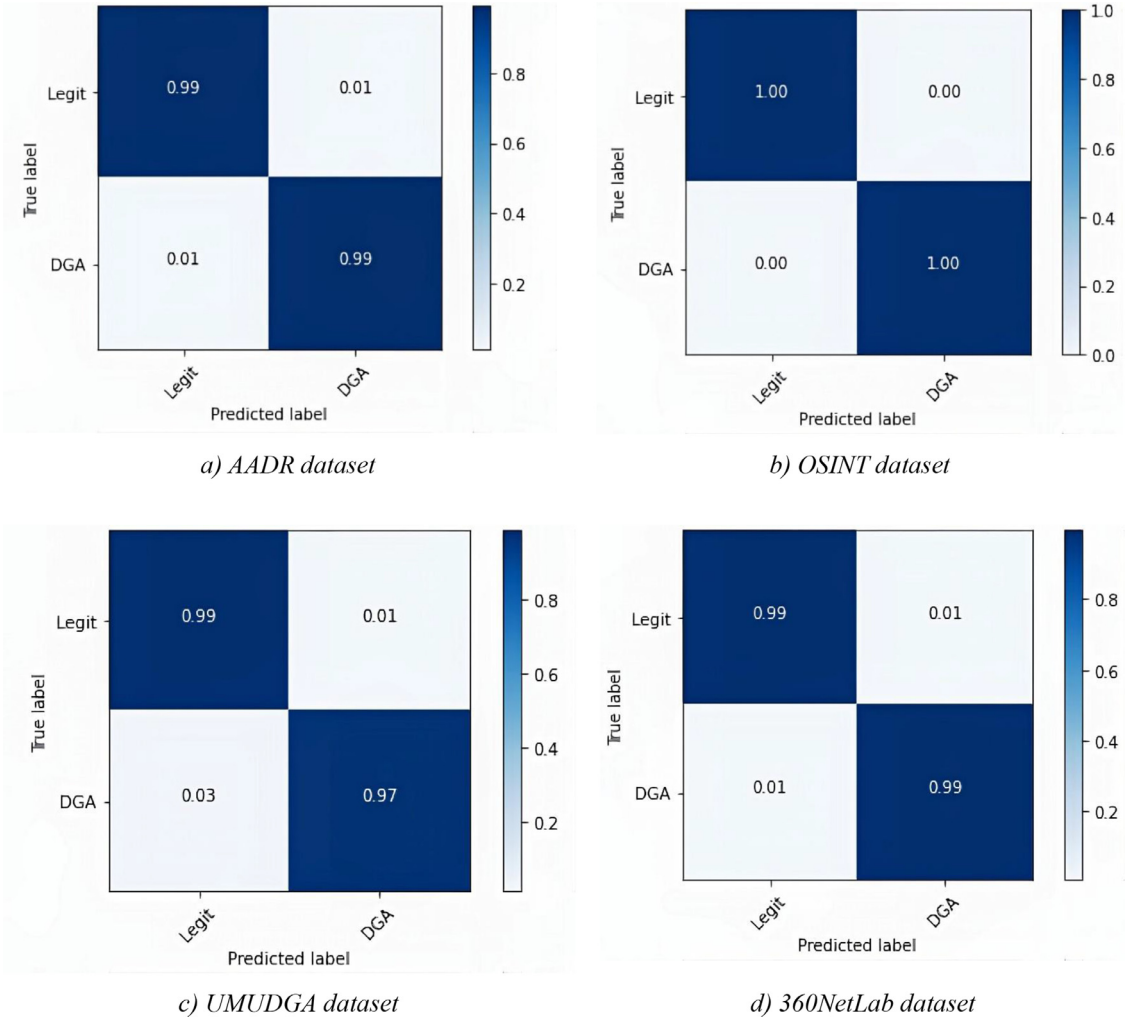


Fig. 11. Confusion matrix corresponding to evaluation the AADR, OSINT, UMUDGA, and 360NetLab datasets.

The results presented above show that the proposed model has very high accuracy in classifying benign and malicious domains, with accuracy reaching 98.32% or more on all 04 datasets. This is also shown through the ROC curve and AUC, with the ROC curve reaching a near-optimal state and the AUC also reaching 0.98 or higher.

In particular, the LA_Bin07 model accurately predicted 100% on the OSINT dataset. This dataset is about the DGA botnets, which are usually being discovered and updated in practice. The result shows that the LA_Bin07 model has all the advantages of the previous algorithm. Also, the slightly lower test results on UMUDGA at 98.32% can be explained by the number of samples and a large number of DGA domain classes included with the 50 families of DGA domains that make up the DGA class.

We also tested the ReLU activation function variations, including the PReLU layer, Leaky ReLu, ELUs, and SoftReLU (SoftPlus). These activation functions gave no better results than the ReLU. Test results show that other variants of ReLU have a longer training time, and the accuracy is not higher. The only advantage is that the testing time is reduced but not significantly.

For a more detailed look at the classification ability for the legit class and the DGA class of the proposed model, the normalized confusion matrix is shown in Fig. 11. It was tested on AADR, OSINT, UMUDGA, and 360NetLab datasets, respectively.

In the matter of ensuring network security against DGA botnets, the misdiagnosis of a benign domain name as malicious can be judged as less dangerous than the omission of a malicious domain name due to mislabeling as a malicious domain name. It shows that the proposed model has a shallow rate of missing DGA botnets.

The proposed model gives the best results on the OSINT set, with a false acceptance and false detection rate of 0.00. The model mistakenly accepted nearly 3% of malicious domains in the opposite direction when evaluated on the UMUDGA dataset. This result can be explained by the UMUDGA suite that has updated several new DGA botnet families, which are very similar to benign domains such as gozi_gpl, gozi_lutherm, gozi_nase, and gozi_rfc4343.

The training and evaluation time of the LA_Bin07 model on the above 04 datasets is shown in Fig. 12:

It can be seen that the LA_Bin07 model has a fast training time, ranging from 3106 s to 6883 s for each dataset. The training time increases in the order of OSINT, UMUDGA, AADR, and 360NetLab with 3106 s, 4424 s, 5083 s, and 6883 s, respectively. The reason is that the number of domain samples included in training also increases in the order above.

In all four experiments, we train the model with epoch = 10. Actual records show that the model starts to converge from epoch = 5 onwards. Moreover, increasing the number of training epochs can slightly improve the accuracy, but it costs a lot in terms of time.

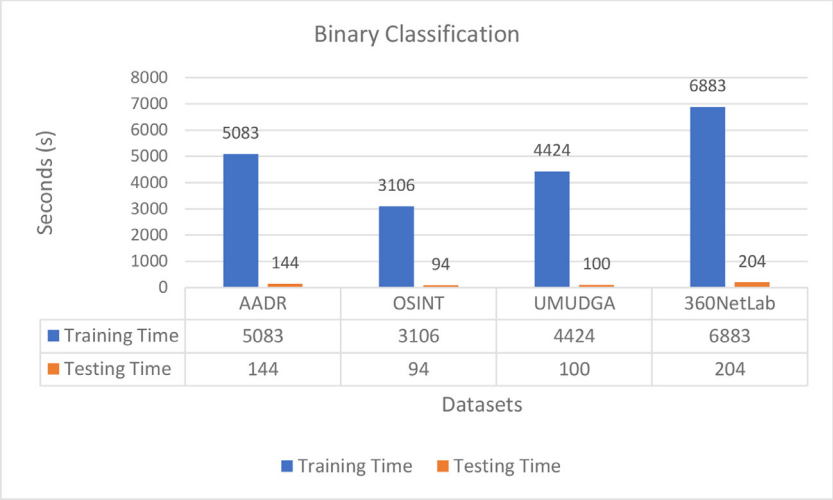


Fig. 12. Training time and testing time of model LA_Bin07.

Table 4
Classification results on the AADR dataset of the model LA_Mul07.

No.	DGA Families	Precision	Recall	F_1-Score
1.	Cryptolocker	1.00	0.98	0.99
2.	Zeus	1.00	1.00	1.00
3.	Pushdo	1.00	1.00	1.00
4.	Rovnix	1.00	1.00	1.00
5.	Tinba	0.98	1.00	0.99
6.	Conficker	1.00	1.00	1.00
7.	Matsnu	1.00	1.00	1.00
8.	Ramdo	1.00	1.00	1.00
Accuracy		99.72%		

Finally, it can be seen that testing time is fast. Therefore, the proposed model can fully meet the requirements of real-time processing in practice.

4.3. Test results of the multiclass classification

We assessed the multiclass classification problem on two datasets: AADR and UMUDGA datasets.

4.3.1. Assess on the AADR dataset

The assessment results of the AADR set are shown in Table 4, Fig. 13, and Fig. 14, respectively, as follows:

The proposed model LA_Mul07 has very high experimental results on the AADR dataset, approaching the level of complete accuracy with the overall accuracy reaching 99.72%. Of the eight considered DGA botnet families, 06 DGA botnet families have detected accurately with Precision, Recall, F1-Score reaching 1.00.

The confusion matrix in Fig. 13 shows that the benign and malicious domains are roughly correctly classified, represented by the value 1.00 in the cells located on the matrix's main diagonal. The cryptolocker family alone has about 2% of the samples misjudged as timba.

Fig. 14 shows the effect of the LA_Mul07 model; with the ROC curve and area lines. It can be seen that the model can achieve high accuracy right at the beginning of the training process. This allows in some time-critical situations, such as discovering a new variant of botnets, the LA_Mul07 model can quickly learn and achieve good enough accuracy to react to that type of botnets before continuing to perfect training for maximum accuracy.

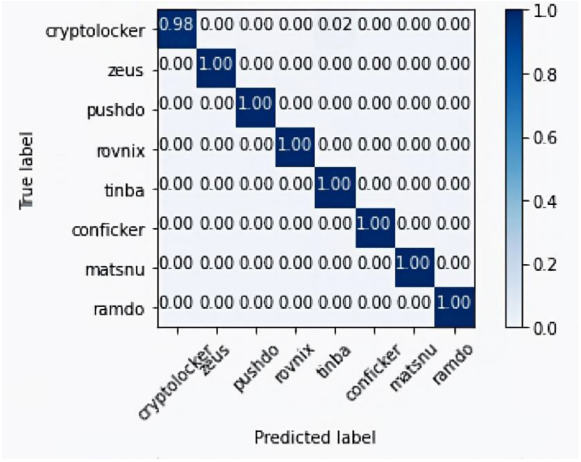


Fig. 13. Confusion matrix when classifying 08 DGA botnet families on the AADR dataset.

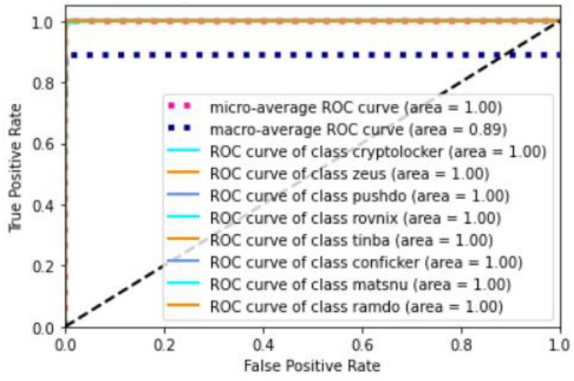


Fig. 14. Representation of ROC curve and AUC on the AADR dataset.

4.3.2. Assess on the UMUDGA dataset

The assessment results of the AADR are shown in Table 5 and Fig. 15, respectively.

Table 5 shows that the proposed LA_Mul07 model has high accuracy in classifying DGA botnets families, even in the case of a large number of DGA botnet families, with an average accuracy of 85.88%. Most of the classes can classify accurately, except for some with relatively low rates, such as alureon, pikspa, pikspa_noise. It

Table 5
Classification results on the UMUDGA set of the model LA_Mul07.

No.	DGA Families	Pre	*Re*	F_1	No.	DGA Families	Pre	*Re*	F_1
1.	alureon	0.45	0.92	0.60	26.	pizd	0.97	0.86	0.91
2.	banjori	0.99	1.00	1.00	27.	proslikefan	0.82	0.65	0.73
3.	bedep	0.96	0.47	0.63	28.	pushdo	0.99	0.99	0.99
4.	ccleaner	1.00	1.00	1.00	29.	pykspa	0.39	0.57	0.47
5.	china	1.00	0.99	1.00	30.	pykspa_noise	0.35	0.16	0.22
6.	corebot	1.00	1.00	1.00	31.	qadars	0.99	0.99	0.99
7.	cryptoloker	0.70	0.66	0.68	32.	qakbot	0.84	0.55	0.67
8.	dircrypt	0.52	0.42	0.47	33.	ramdo	1.00	1.00	1.00
9.	dyre	1.00	1.00	1.00	34.	ramnit	0.44	0.66	0.52
10.	fobber_v1	0.88	1.00	0.93	35.	ranbyus_v1	0.76	0.98	0.86
11.	fobber_v2	0.48	0.08	0.14	36.	ranbyus_v2	0.76	0.88	0.82
12.	gozi_gpl	0.96	0.99	0.98	37	rovnix	0.97	0.94	0.95
13.	gozi_luther	0.97	0.95	0.96	38	shiotob	1.00	0.90	0.95
14.	gozi_nase	0.89	0.97	0.93	39	simda	1.00	1.00	1.00
15.	gozi_rfc4343	0.91	0.98	0.90	40	aaron	1.00	1.00	1.00
16.	kraken_v1	0.72	0.96	0.83	41	suppobox_1	0.87	0.97	0.92
17.	kraken_v2	0.82	0.41	0.55	42	suppobox_2	0.98	1.00	0.99
18.	locky	0.84	0.62	0.71	43	suppobox_3	0.99	1.00	1.00
19.	matsnu	0.98	0.94	0.96	44	symmi	1.00	1.00	1.00
20.	murofet_v1	0.99	1.00	1.00	45	tempedreve	0.58	0.86	0.69
21.	murofet_v2	0.94	0.96	0.95	46	tinba	0.77	0.97	0.86
22.	murofet_v3	1.00	1.00	1.00	47	vawtrak_v1	1.00	1.00	1.00
23.	necurs	0.99	0.80	0.89	48	vawtrak_v2	0.99	1.00	1.00
24.	maim	0.95	0.94	0.95	49	vawtrak_v3	1.00	1.00	1.00
25.	padcrypt	1.00	1.00	1.00	50	zeus_newgoz	1.00	1.00	1.00
Accuracy					85.88%				

can be solved by separating and training these classes separately. In general, with classifying 50 classes of the UMUDGA dataset, our proposed model gives very satisfactory results.

UMUDGA dataset consists of 50 DGA botnet families corresponding to 50 classes. Fig. 15 shows the confusion matrix of the LA_Mul07 model with a bold main diagonal, showing the model's accuracy. We can also draw out the remaining problems, including the fobber_v2 family misclassified as alureon with the rate of 90%; about 40% of the kraken_v2 family is misclassified kreken_v1; about 60% of the pykspa_noise family is misclassified as pykspa. This can be explained by two-thirds of these families being improved variants, which can confuse the classifier. Assessment on the remaining DGA botnet families, the LA_Mul07 classifier gives high accuracy.

The training and evaluation times in the multiclass classification problem on two datasets AADR and UMUDGA dataset, are shown in Fig. 16.

It can be seen that the model LA_Mul07 has a fast training time in the experiments. With 2324 s and 1339 s on two data sets, AADR and UMUDGA, respectively, because of the larger number of AADR training samples.

We also find that the number of classes of each dataset does not affect the training time much. The UMUDGA dataset has the number of classes to be classified as 50, much more than the AADR set of 08 classes. This result shows that the LA_Mul07 architecture has the added advantage of extending the detection of more DGA botnets families as needed without requiring an increase in model training time.

Regarding the test time, similar to the LA_Bin07 model, the LA_Mul07 model has a fast test time on the test data set, which allows the model to be applied in practice to solve the required problems quickly-processing requests in real-time.

5. Assessment with relevant studies

5.1. Assessment with other studies on the UMUDGA dataset

The UMUDGA dataset was built by Zago et al. (2020) at the University of Murcia, Spain. This dataset can be considered as the new and the most complete dataset on the DGA botnets, with 50 families of DGA botnets aggregated and classified.

In Zago et al. (2020) used machine learning models including AdaBoost (AB), Neural Network (NN), Random Forest (RF), Support Vector Machines (SVM - using C = 1.00, $\epsilon = 0.10$, and RBF Kernel), Decision Tree (DT), and k-Nearest Neighborhood (kNN) to solve the above binary and multiclass classification problem. We compare and evaluate the proposed model LSTM_07 with the models mentioned in the study above. The test environment and the number of samples are similar. In addition to Zago et al. (2020), not many other studies on DGA botnet detection have been assessed on this dataset.

For the binary classification task, the comparison results are summarized and shown in Fig. 17.

The results from Fig. 17 show that the LA_Bin07 model has much better accuracy than the SVM model. At the same time, the results are almost equivalent to the AB, NN, RF, DT, or kNN models, with Accuray achieving 98%–99%. The experiment shows that the model LA_Bin07 gives outstanding results.

For the multiclass classification task, the comparison results are summarized and shown in Fig. 18.

For the multiclass classification problem, the accuracy of LA_Mul07 model is much higher than the rest of the machine learning models. Accuracy, Precision, Recall, and F1-score reached 86%, 87%, 86%, and 85%, respectively. Meanwhile, NN's second-best model is only 77%, 78%, 77%, and 77%, respectively. While achieving very high accuracy in the binary classification problem, the kNN model proved inaccurate in the multiclass classification problem. It can be concluded that the LA_Mul07 model gives more accurate multiclass classification results than other models.

5.2. Assessment with other deep learning models in multiclass classification

The binary classification problem has been considered by many studies, with accuracy achieving the highest from 98%–99%. In this section, we consider the problem of multiclass classification in more detail. A multiclass classification model called LSTM.MI was proposed by Duc et al. in Tran et al. (2018), besides other

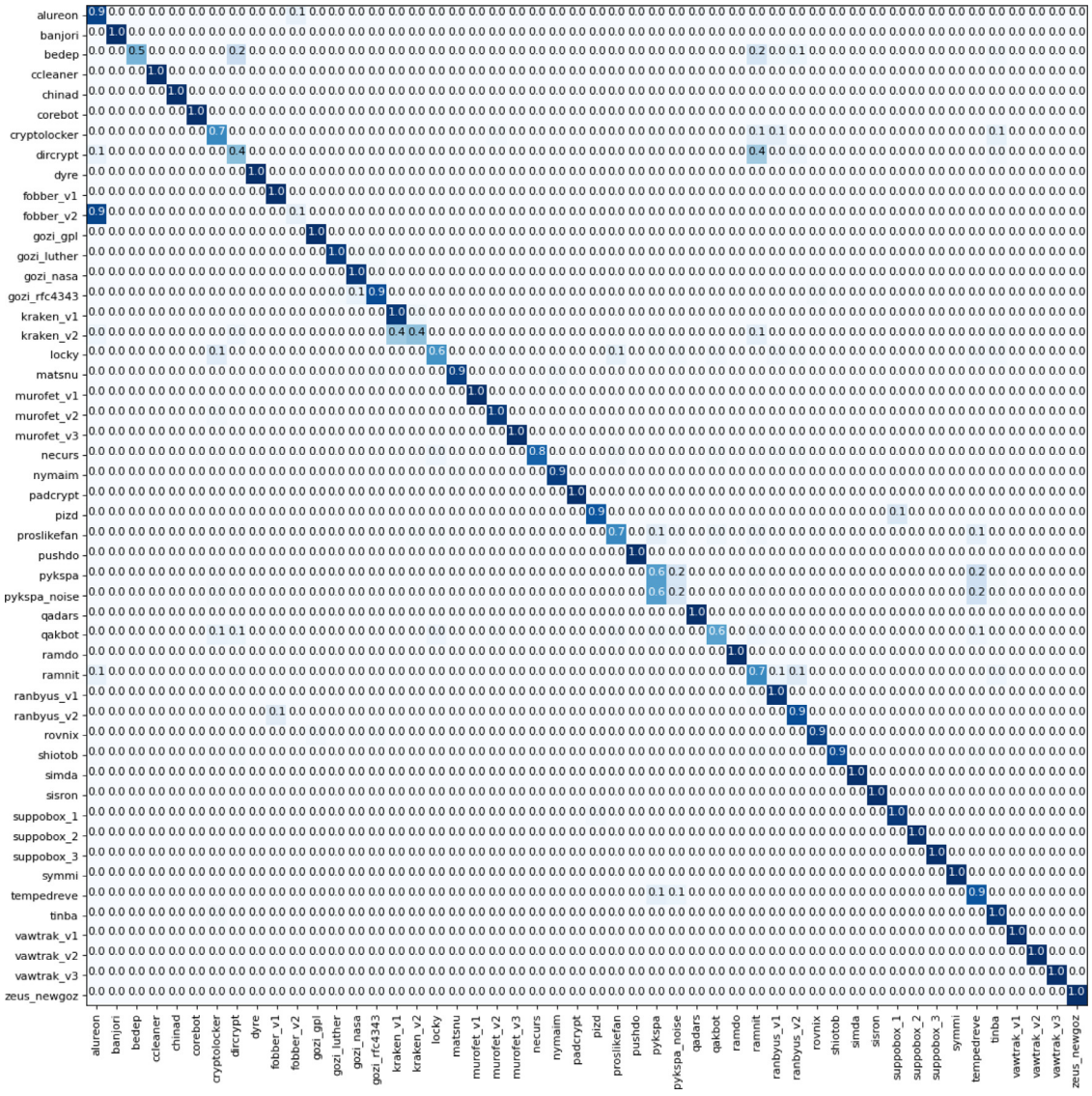


Fig. 15. Confusion matrix for multiclass classification on the UMUDGA dataset.

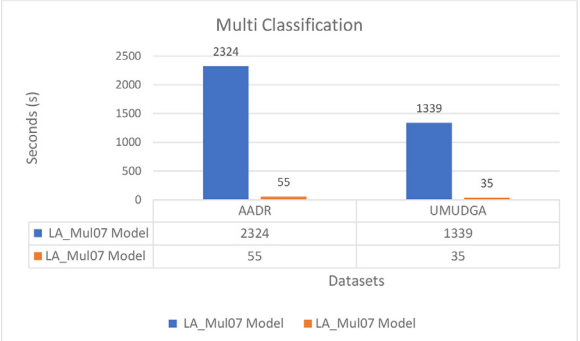


Fig. 16. Training time and testing time of model LA_Mul07.

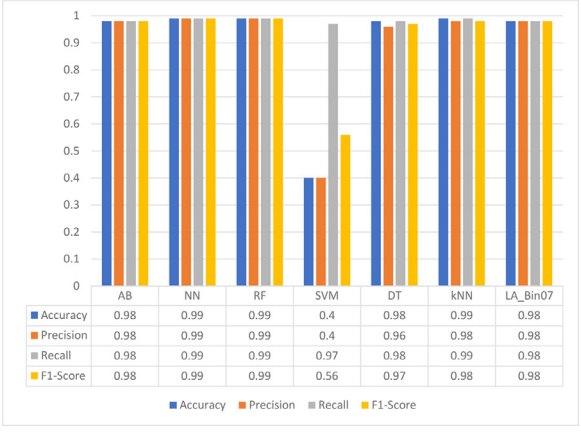
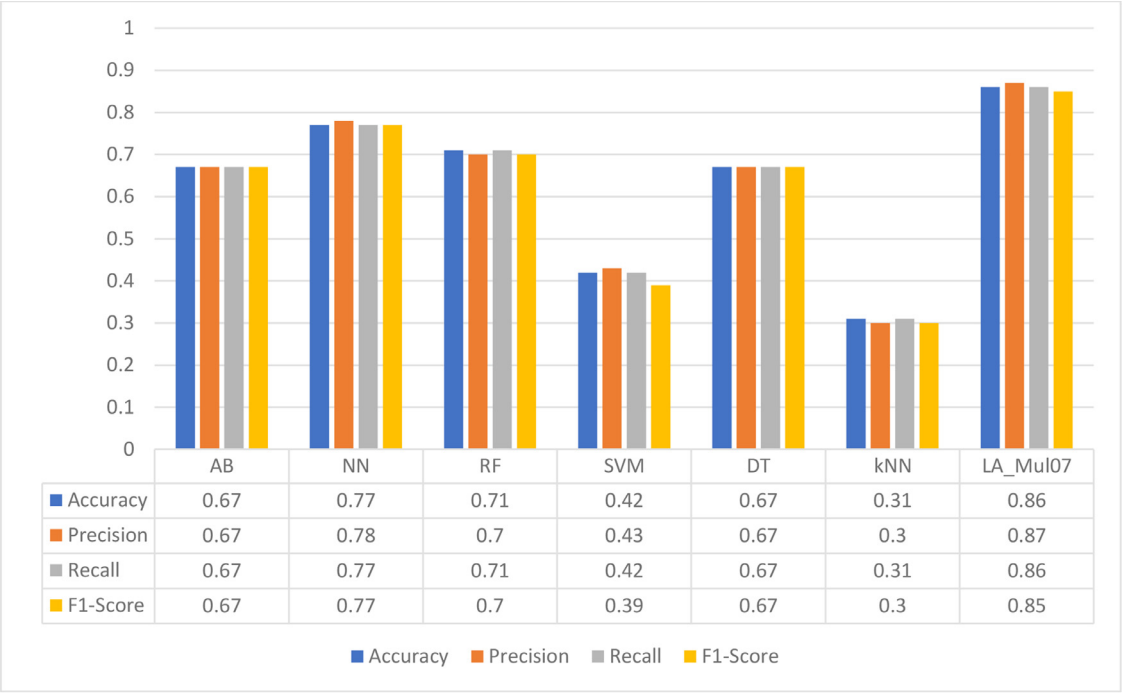


Fig. 17. Comparison of binary classification results on the UMUDGA dataset.

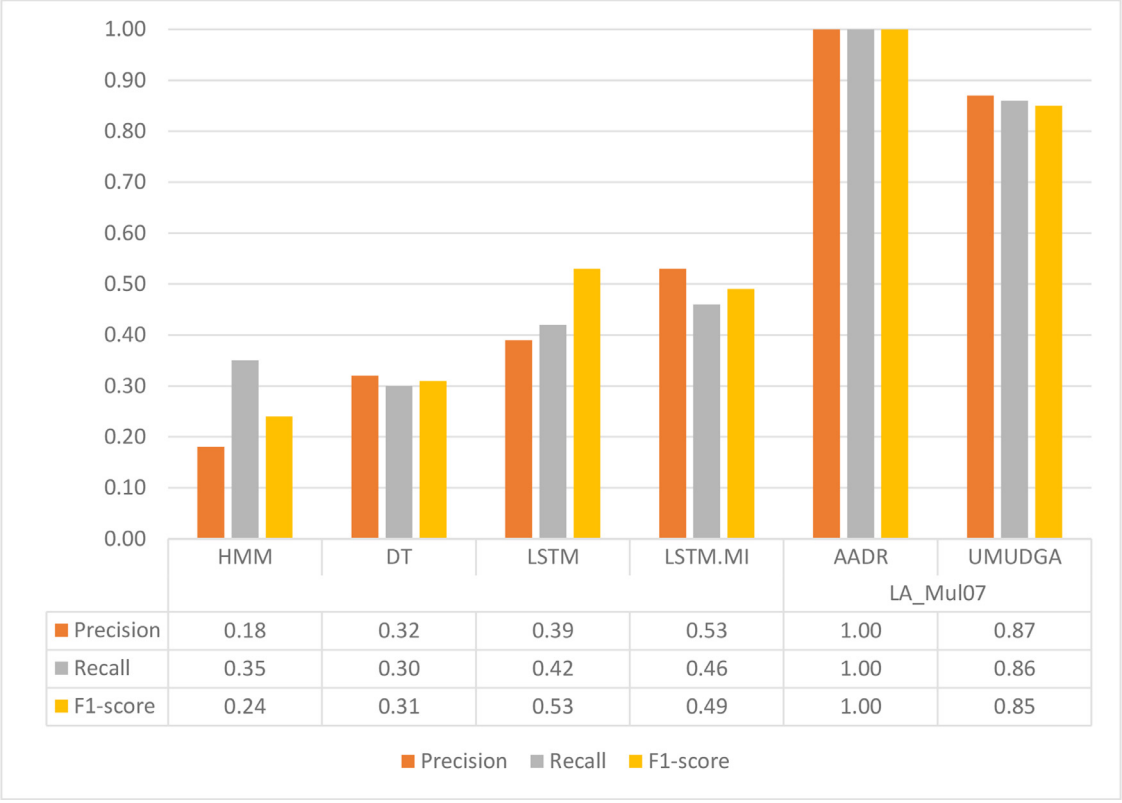
models include Hidden Markov Model (HMM) (Antonakakis et al., 2012), Decision Tree (DT) (Quinlan, 2004), and the original LSTM (LSTM) (Woodbridge et al., 2016). The reason we chose this study is because of the similarity in the test environment and data set. Duc et al. (Tran et al., 2018) used a combination of the OSINT set and the dataset collected from the study (Woodbridge et al., 2016), with 37 DGA botnet families synthesized.

The results of the comparison between the models are presented in Fig. 19. It can be seen that the proposed LA_Mul07 model gives much higher Precision, Recall, and F1-Score results than the HMM, DT, LSTM, and LSTM.MI. The new model has improved ac-



	AB	NN	RF	SVM	DT	kNN	LA_Mul07
Accuracy	0.67	0.77	0.71	0.42	0.67	0.31	0.86
Precision	0.67	0.78	0.7	0.43	0.67	0.3	0.87
Recall	0.67	0.77	0.71	0.42	0.67	0.31	0.86
F1-Score	0.67	0.77	0.7	0.39	0.67	0.3	0.85

Fig. 18. Comparison of multiclass classification results on the UMUDGA dataset.



	HMM	DT	LSTM	LSTM.MI	AADR	UMUDGA LA_Mul07
Precision	0.18	0.32	0.39	0.53	1.00	0.87
Recall	0.35	0.30	0.42	0.46	1.00	0.86
F1-score	0.24	0.31	0.53	0.49	1.00	0.85

Fig. 19. Assessment results for the binary classification problem between models.

curacy much compared to the traditional LSTM model or LSTM.MI that was proposed earlier.

The results show that the LA_Mul07 model has higher Precision, Recall, and F1-Score than the matched models, and the remaining models have much lower results. The reason is (1) The HMM, DT, LSTM, LSTM.MI models use a pure algorithm or have a simple design model, leading to the training process may miss information. (2). HMM, DT, LSTM, LSTM.MI models are almost undetectable for some DGA botnet families such as Cryptowall, Matsnu, and Suppobox, with precision being almost 0%. While the model LA_Mul07 does not have got the above problem (Table 5). (3). The LA_Mul07 model uses an additional layer of attention to evaluate the weight

of the input information, so the model will select the information that needs to be trained more accurately.

It should be noted that the LA_Mul07 model is also evaluated on the UMUDGA dataset consisting of 50 DGA Domain families, which means that it has higher detection accuracy on many DGA botnet families than the previous LSTM.MI model (evaluated on 37 DGA botnet families). For the AADR dataset, the LA_Mul07 model gives almost entirely accurate results.

6. Conclusion and future works

In this study, we presented our research on DGA botnets. To solve the binary and multiclass classification problems, we have proposed two new deep learning models: LA_Bin07 and LA_Mul07. The binary classification model solves the problem of classifying benign and malicious domains, whereas the multi-layer classification model identifies the DGA botnets family among the detected malicious domains.

The assessment was conducted on four datasets, including AADR, OSINT, UMUDGA, and 360NetLab. The assessment results show that the LA_Bin07 model effectively solves the binary classification problem with very high accuracy. In the multiclass classification problem, the LA_Mul07 model achieved 99.72% and 85.88% accuracy on two datasets: AADR and UMUDGA, respectively. These are much more significant improvements than the previously proposed models.

Our research has proposed two new models, LA_Bin07 and LA_Mul07, which have fully solved two critical problems, including botnet detection and botnets family classification. Experiments show that the proposed model has improved accuracy compared to previous models. This is especially evident in the multiclass classification problem when the accuracy is greatly improved. Besides, the experiments were conducted on reliable and widely published data sets. Specifically, the UMUDGA dataset published in 2020 is something that most previous studies have not evaluated. Finally, the training and testing time of the model is fast, which can completely meet the requirements of fast processing for natural systems.

Although achieving high accuracy in the multiclass classification problem, our LA_Mul07 model still has a limitation: the accuracy of some families of botnets is still not high. Some botnets with multiple versions can also appear confusing. In terms of the current development, the team continues to do in-depth research on DGA botnets, improving the accuracy of the multiclass classification model based on more profound training on classes with a high confusion rate. We are going to use the new hybrid models or voting-based models to correctly classify cases where the distinction between classes is not apparent. We will also conduct model experiments in real networks with high traffic to evaluate the performance and accuracy of the algorithm when operating in the real network.

Declaration of Competing Interest

The authors declare that:

They have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Tong Anh Tuan: Software, Investigation, Visualization, Writing – original draft, Writing – review & editing. **Hoang Viet Long:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision. **David Taniar:** Writing – review & editing, Methodology.

Acknowledgments

Tong Anh Tuan was funded by Vingroup Joint Stock Company and supported by the Domestic Master/ PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA), code VINIF.2020.TS.106. This research is funded by National Science and Technology Major Project of the Ministry of Science and Technology of Vietnam under grant number ĐTĐLCN.105/21-C.

References

A. Abakumov, "DGA Repository," GitHub, 2016. https://github.com/andrewaeva/DGA (accessed Jun. 08, 2021).

Alauthman, M., Aslam, N., Al-kasassbeh, M., Khan, S., Al-Qerem, A., Raymond Choo, K.K., 2020. An efficient reinforcement learning-based Botnet detection approach. J. Netw. Comput. Appl. 150, 102479. doi:10.1016/j.jnca.2019.102479.

Alexa Internet Inc., 2019. Alexa Top 1 Million Sites. Kaggle Datasets http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.

Anderson, R., et al., 2019. Measuring the changing cost of cybercrime our framework for analysing the costs of cybercrime. Work. Econ. Inf. Secur. 1–32. [Online] Available http://orca.cf.ac.uk/id/eprint/122684.

Antonakakis, M., et al., 2012. From throw-away traffic to bots: detecting the rise of DGA-based malware. In: Proceeding 21st USENIX Security Symposium, pp. 491–506.

Bilge, L., Sen, S., Balzarotti, D., Kirda, E., Kruegel, C., 2014. EXPOSURE: a passive DNS analysis service to detect and report malicious domains. ACM Trans. Inf. Syst. Secur. 16 (4). doi:10.1145/2584679.

Bisio, F., Saeli, S., Lombardo, P., Bernardi, D., Perotti, A., Massa, D., 2017. Real-time behavioral DGA detection through machine learning. In: International Carnahan Conference on Security Technology, pp. 1–6. doi:10.1109/CCST.2017.8167790 2017-Octob.

Chen, Z. and Subramanian, D. "An Unsupervised approach to detect spam campaigns that use botnets on twitter," 2018, [Online]. Available: http://arxiv.org/abs/1804.05232.

Chowdhury, S., et al., 2017. Botnet detection using graph-based feature clustering. J. Big Data 4 (1). doi:10.1186/s40537-017-0074-7.

Curtin, R.R., Gardner, A.B., Grzonkowski, S., Kleymenov, A., Mosquera, A., 2019. Detecting DGA domains with recurrent neural networks and side information. ACM International Conference Proceeding Series doi:10.1145/3339252.3339258.

Du, K.L., Swamy, M.N.S., 2014. Recurrent neural networks. In: Neural Networks and Statistical Learning, pp. 337–353. doi:10.1007/978-1-4471-5571-3_11.

Eckle, K., Schmidt-Hieber, J., 2019. A comparison of deep networks with ReLU activation function and linear spline-type methods. Neural Netw. 110, 232–242. doi:10.1016/j.neunet.2018.11.005.

E. U. A. for C. (ENISA), "Major DDoS attacks involving IoT devices," [Online], 2019. available:%0Ahttps://www.enisa.europa.eu/publications/info- notes/major-ddos-attacks-involving-iot-devices. (accessed Jun. 16, 2021).

Gao, B. and Pavel, L. "On the properties of the softmax function with application in game theory and reinforcement learning," arXiv, 2017.

Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., Srivastava, N., 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15 (1), 1929–1958. [Online] Available http://jmlr.org/papers/v15/srivastava14a.html.

Hochreiter, S., 1996. Lstm can solve hard long time lag problems. Advances in Neural Information Processing Systems.

Ianelli, N., Hackworth, A., 2007. Botnets as a vehicle for online crime. Int. J. Forensic Comput. Sci. 19–39. doi:10.5769/j200701002.

John, J.P., Moshchuk, A., Gribble, S.D., Krishnamurthy, A., 2009. Studying spamming botnets using Botlab. In: Proceeding 6th USENIX Symp. Networked Syst. Des. Implementation, NSDI 2009, pp. 291–306.

Karim, A., Bin Salleh, R., Shiraz, M., Shah, S.A.A., Awan, I., Anuar, N.B., 2014. Botnet detection techniques: review, future trends, and issues. J. Zhejiang Univ. Sci. C 15 (11), 943–983. doi:10.1631/jzus.C1300242.

Khan, W.Z., Khan, M.K., Bin Muhaya, F.T., Aalsalem, M.Y., Chao, H.C., 2015. A comprehensive study of email spam botnet detection. IEEE Commun. Surv. Tutor. 17 (4), 2271–2295. doi:10.1109/COMST.2015.2459015.

Khan, R.U., Zhang, X., Kumar, R., Sharif, A., Golilarz, N.A., Alazab, M., 2019. An adaptive multi-layer botnet detection technique using machine learning classifiers. Appl. Sci. 9 (11). doi:10.3390/app9112375.

Kim, W., Jeong, O.R., Kim, C., So, J., 2011. The dark side of the Internet: attacks, costs and responses. Inf. Syst. 36 (3), 675–705. doi:10.1016/j.is.2010.11.003.

Kwon, J., Lee, J., Lee, H., Perrig, A., 2016. PsyBoG: a scalable botnet detection method for large-scale DNS traffic. Comput. Netw. 97, 48–73. doi:10.1016/j.comnet.2015.12.008.

Li, J., Liu, X., Jin, J. and Yu, S. "Too expensive to attack: a joint defense framework to mitigate distributed attacks for the internet of things grid," 2021, [Online]. Available: http://arxiv.org/abs/2104.00236.

Mac, H., Tran, D., Tong, V., Nguyen, L.G., Tran, H.A., 2017. DGA botnet detection using supervised learning methods. In: ACM International. Conference Proceeding Series, pp. 211–218. doi:10.1145/3155133.3155166 2017-Decem.

360NetLab, 2016. DGA – Netlab Opendata Project. Qihoo 360 Technology http://data.netlab.360.com/dga/.

Nguyen, T.D., Cao, T.D., Nguyen, L.G., 2015. DGA botnet detection using collaborative filtering and density-based clustering. ACM Internatioal Conference Proceeding Series 203–209. doi:10.1145/2833258.2833310, 03-04-Dece.
Nguyen, H.T., Ngo, Q.D., Le, V.H., 2020. A novel graph-based approach for IoT botnet detection. Int. J. Inf. Secur. 19 (5), 567–577. doi:10.1007/s10207-019-00475-6.
Nicholson, Paul, 2021. Attack Shows it is Imperative for a DDoS Zero-Trust Approach and Continued Diligence. A10 Networks https://www.a10networks.com/blog/aws-hit-by-largest-reported-ddos-attack-of-2-3-tbps/.
Nugraha, B., Nambiar, A., Bauschert, T., 2020. Performance evaluation of botnet detection using deep learning techniques. In: Proceeding 11th International. Conference Network Future NoF 2020, pp. 141–149. doi:10.1109/NoF50125.2020.9249198.
OSINT, "Feeds from bambenek consulting," 2021. https://osint.bambenekconsulting.com/feeds/ (accessed Mar. 15, 2021).
Pa, Y.M.P., Suzuki, S., Yoshioka, K., Matsumoto, T., Kasama, T., Rossow, C., 2016. IoTPOT: a novel honeypot for revealing current IoT threats. J. Inf. Process. 24 (3), 522–533. doi:10.2197/ipsjjip.24.522.
Putman, C.G.J., Abhishta, A., Nieuwenhuis, L.J.M., 2018. Business model of a botnet. In: Proceeding - 26th Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2018, pp. 441–445. doi:10.1109/PDP2018.2018.00077.
Quinlan, J.R. "Data mining tools See 5 and C5. 0.," 2004.
Rahbarinia, B., Perdisci, R., Lanzi, A., Li, K., 2013. PeerRush: mining for unwanted P2P traffic. Lect. Notes Comput. Sci. 7967, 62–82. doi:10.1007/978-3-642-39235-1_4, (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)LNCS.
Saad, S., et al., 2011. Detecting P2P botnets through network behavior analysis and machine learning. In: 2011 9th Annual. International. Conference Privacy, Security Trust. PST 2011, pp. 174–180. doi:10.1109/PST.2011.5971980.
Schiller, C., et al., 2007. Botnets: The Killer Web Applications. Syngress Publ, pp. 29–75.
Sharifnya, R., Abadi, M., 2015. DFBotKiller: domain-flux botnet detection based on the history of group activities and failures in DNS traffic. Digit. Investig. 12, 15–26. doi:10.1016/j.diin.2014.11.001.
Sharma, Siddharth, Sharma, Simone, Athaiya, Anidhya, 2020. Activation functions in neural networks. Int. J. Eng. Appl. Sci. Technol. 4 (12), 310–316. [Online] Available http://www.ijeast.com.
Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A.A., 2012. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Comput. Secur. 31 (3), 357–374. doi:10.1016/j.cose.2011.12.012.
Sibi, P., Jones, S.A., Siddarth, P., 2013. Analysis of different activation functions using back propagation neural networks. J. Theor. Appl. Inf. Technol. 47 (3), 1344–1348.
Simran, K., Balakrishna, P., Vinayakumar, R., Soman, K.P., 2020. Deep learning based frameworks for handling imbalance in DGA, Email, and URL Data Analysis. Commun. Comput. Inf. Sci. 1213, 93–104. doi:10.1007/978-981-15-9700-8_8.
Suryotrisongko, H. "Computable CTI: sharing AI model for the next level of actionable cyber threat intelligence. case study of botnet detection," IEEE Open Access J., 2020.
Tran, D., Mac, H., Tong, V., Tran, H.A., Nguyen, L.G., 2018. A LSTM based framework for handling multiclass imbalance in DGA botnet detection. Neurocomputing 275, 2401–2413. doi:10.1016/j.neucom.2017.11.018.
Tuan, T.A., Long, H.V., Son, L.H., Kumar, R., Priyadarshini, I., Son, N.T.K., 2020. Performance evaluation of Botnet DDoS attack detection using machine learning. Evol. Intell. 13 (2), 283–294. doi:10.1007/s12065-019-00310-w.
Tuptuk, N., Hailes, S., 2018. Security of smart manufacturing systems. J. Manuf. Syst. 47, 93–106. doi:10.1016/j.jmsy.2018.04.007.
Ullah, I., Khan, N., Aboalsamh, H.A., 2013. Survey on botnet: its architecture, detection, prevention and mitigation. In: 2013 10th IEEE Int. Conf. Networking, Sens. Control. ICNSC 2013, pp. 660–665. doi:10.1109/ICNSC.2013.6548817.
Van Can, N., Tu, D.N., Tuan, T.A., Long, H.V., Son, L.H., Son, N.T.K., 2020. A new method to classify malicious domain name using neutrosophic sets in DGA botnet detection. J. Intell. Fuzzy Syst. 38 (4), 4223–4236. doi:10.3233/jifs-190681.
Vaswani, I., Shazeer, A., Parmar, N., Uszkoreit, N., Jones, J., Gomez, L., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 5998–6008.
"VirusShare.com - Because Sharing is Caring." https://virusshare.com/ (accessed Mar. 15, 2021).
Wang, J., Paschalidis, I.C., 2017. Botnet Detection based on anomaly and community detection. IEEE Trans. Control Netw. Syst. 4 (2), 392–404. doi:10.1109/TCNS.2016.2532804.
Wang, T.S., Lin, H.T., Cheng, W.T., Chen, C.Y., 2017. DBod: clustering and detecting DGA-based botnets using DNS traffic analysis. Comput. Secur. 64, 1–15. doi:10.1016/j.cose.2016.10.001.
Woodbridge, J., Anderson, H.S., Ahuja, A. and Grant, D. "Predicting domain generation algorithms with long short-term memory networks," 2016, [Online]. Available: http://arxiv.org/abs/1611.00791.
Yavanoglu, O., Aydos, M., 2017. A review on cyber security datasets for machine learning algorithms. In: Proceeding - 2017 IEEE International. Conference Big Data, Big Data 2017, pp. 2186–2193. doi:10.1109/BigData.2017.8258167 2018-Janua,.
Zago, M., Pérez, M.Gil, Martínez Pérez, G., 2020a. UMUDGA: a dataset for profiling algorithmically generated domain names in botnet detection. Data Br 30, 105400. doi:10.1016/j.dib.2020.105400.
Zago, M., Pérez, M.Gil, Martínez Pérez, G., 2020b. UMUDGA: a dataset for profiling DGA-based botnet. Comput. Secur. 92. doi:10.1016/j.cose.2020.101719.
Zhou, Y.-.L., Li, Q.-.S., Miao, Q., Yim, K., 2013. DGA-Based botnet detection using DNS traffic. J. Internet Serv. Inf. 3 (11), 116–123. [Online] Available http://isyou.info/jisis/vol3/no34/jisis-2013-vol3-no34-11.pdf.

Tong Anh Tuan is the PhD student in Information Security at Vietnam Academy of Sciences and Technology. He now serves as the lecturer in Faculty of Information Technology, University of Technology-Logistics of Public Security. He currently concerns about attack and defense techniques in network security and privacy.

Hoang Viet Long received PhD diploma in Computer Science at Hanoi University of Science and Technology in 2011, specializing in fuzzy computing and soft computing techniques with applications to electronic engineering. He has been promoted to Associate Professor since 2018. Recently, he has been concerned about Machine Learning with applications to Cybersecurity. .

David Taniar received the bachelor's, master's, and Ph.D. degrees all in computer science, specializing in databases. His research areas are in big data processing, data warehousing, and mobile and spatial query processing. He has published a book on the High-Performance Parallel Database Processing (Wiley, 2008). He has also published over 200 journal papers. He is a regular keynote speaker at an international conference, delivering lectures and speeches on big data. He is a founding Editor-in-Chief of the International Journal of Web and Grid Services, and the International Journal of Data Warehousing and Mining. He is currently an Associate Professor with Monash University, Australia.