A DGA Domain Name Detection Model Based on A Hybrid Deep Neural Network with Multi-dimensional Features

Rui Pan, Yu Wang*, Zuchao Wang

Abstract—**Effective detection of DGA (Domain Generation Algorithm) domain names is crucial for identifying and countering Botnets, and safeguarding cyber security. In this paper, we propose a new detection method using a hybrid deep neural network with multi-dimensional features. Firstly, multi-dimensional features are employed to bolster extracting the implicit semantic content inherent in DGA domain names. Secondly, a hybrid deep neural network, which integrates both CNN (Convolutional Neural Network) and BiLSTM (Bi-directional Long Short-Term Memory network), is utilized to effectively extract and synthesize the distinctive features of DGA domain names. Finally, comparison experiments are designed to evaluate the model's overall performance and detection accuracy. Experimental results demonstrate the efficacy of the proposed model. In the two-classification, we attained a precision rate of 97.72% and an impressive F1 score of 98.20%, indicative of a fine balance between precision and recall. In the multi-classification, our model still performed well, with a precision rate of 96.90% and an F1 score of 96.92%, further underscoring its robustness and adaptability. Compared to other models, our model achieved a detection rate of 100% for more DGA families. The model demonstrated powerful abilities, especially in distinguishing among different semantic features, and it exhibited particularly exceptional detection performance for DGA domain names generated with fixed lengths or fixed letter patterns.**

Index Terms—**DGA domain names, word embedding, CNN, BiLSTM, semantic feature, deep learning, cyber security**

I. INTRODUCTION

BOTNETS pose a significant threat to cyber security. Attackers exploit software vulnerabilities and various techniques to infiltrate malicious zombie programs, worms, or viruses into numerous target host systems by the one-to-many C&C (command and control) server [1],[2].

Upon receiving directives from the C&C server, the zombie hosts are capable of executing a range of nefarious activities, such as launching DDoS (Distributed Denial of Service) attacks, disseminating spam emails, and transmitting Trojan horse viruses [3],[4].

In the Domain-Flux mechanism, domain names generated by generation algorithms are called DGA (Domain Generation Algorithm) domain names. By using DGA domain names, the Domain-Flux mechanism effectively sustains communication between the C&C server and bots, exhibiting robust anti-interference capabilities that thwart detection by security systems [5].

Effective detection of DGA domain names is crucial in cybersecurity, enabling precise and timely identification of Botnets and alerting users to potential threats.

Currently, detection methods are mainly bifurcated into two principal categories. One approach is predicated on the correlation characteristics of domain names, and DNS resolution information such as IP and traffic data of the domain name system is needed. However, this method entails a significant consumption of resources and time [6].

The alternative approach focuses on the character features inherent in domain names. Since domain names are inherently constructed from characters, the information encapsulated within these characters can be harnessed in DGA domain name detection. This method offers the advantage of real-time detection and is straightforward to implement. Scholars have introduced a diverse array of detection techniques that focus on the character-based analysis of DGA domain names.

Initially, DGA domain name detection predominantly relied on traditional machine learning algorithms. However, these algorithms posed challenges in terms of memory consumption and computational time during the training process, making their implementation for large-scale samples particularly difficult [7].

Deep learning has been advancing at an unprecedented pace, and its application in DGA domain name detection has seen significant progress. This paper studied the DGA domain names detection method and proposed a hybrid deep learning model (CNN-BiLSTM-M).

By incorporating multi-dimensional features and hybrid neural networks, classification accuracy in DGA domain name detection could be improved and the generalization ability of the detection model is enhanced.

In summary, contributions of the paper include the following:

1. Improving word embedding method.

In this paper, we delve deeply into the character-level word embedding methodology and extend it by innovatively incorporating semantic features. This method can significantly enhance the extraction of latent information, particularly semantic nuances, within domain names.

2. Improving the feature extraction and fusion ability of the deep learning model.

We introduce a hybrid deep neural network model that integrates CNN (Convolutional Neural Network) and

Manuscript received April 9, 2024; revised November 26, 2024.

Rui Pan is a senior engineer at China Academy of Information and Communications Technology, Beijing 100191, China. (e-mail: pan-rui@foxmail.com).

Yu Wang is an assistant engineer at CATARC（Tianjin）Automotive Engineering Research Institute Co.,Ltd, Tianjin, 300399, China. (corresponding author to provide phone: +86-18847163202; e-mail: 18847163202@163.com).

Zuchao Wang is a professor at the China University of Geosciences (Beijing), Beijing 100083, China. (e-mail: wzc@cugb.edu.cn).

BiLSTM (Bi-directional Long Short-Term Memory). The model excels at capturing intricate local phrase features alongside bidirectional global dependencies across multiple dimensions.

3. In the experimental section, we designed comparison experiments.

The detection performance of the model was evaluated by conducting contrast experiments. In the two-classification scenario, we attained a precision rate of 97.72% and an impressive F1 score of 98.20%. In the multi-classification scenario, our model still performed well, with a precision rate of 96.90% and an F1 score of 96.92%.

The paper is divided into 6 chapters. In Chapter 1, we present background knowledge on DGA domain name detection. In Chapter 2, we summarize the current work. In Chapter 3, we describe the multi-dimensional features and the proposed model. Chapter 4 describes the data set and experiment design in detail. The analysis of the experimental results is presented in Chapter 5. Finally, we conclude the paper and discuss possible future works in Chapter 6.

II. Related Work

The DGA domain-generating process is shown in Figure 1. SLD (Second-level domain) is generated by different algorithms and random seeds (which can be obtained from public resources). Subsequently, DGA domain names can be generated by concatenating SLD with TLD (Top-level domain).

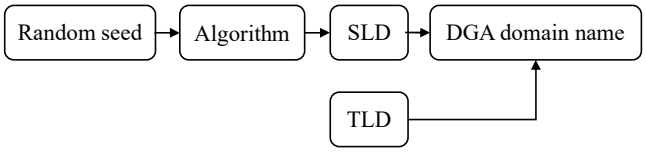

Fig. 1. DGA domain generating process

There are many types of DGA domain names, and their survival periods are mainly between 1 and 7 days. According to different generation algorithms, DGA domain names are classified as follows:

TABLE I
DGA Domain Names Sorted By Algorithm

DGA domain name	Algorithm description	DGA family
Arithmetic-based DGA domain name	Numeric sequences can be obtained through arithmetic operations, for example: using ASCII codes to generate domain names or as offsets pointed to hard-coded character tables.	e.g. banjori, conficker
Hash-based DGA domain name	Using the hexadecimal value of hash value, which can be obtained through MD5, SHA 256, etc., to generate domain names.	e.g. bamital, dyre
Wordlist-based DGA domain name	Selecting words from the word list and combining them into domain names	e.g. matsnu, suppobox
Replacement-based DGA domain name	Replacing the initial domain name to obtain all possible domain names.	e.g. volatilece dar

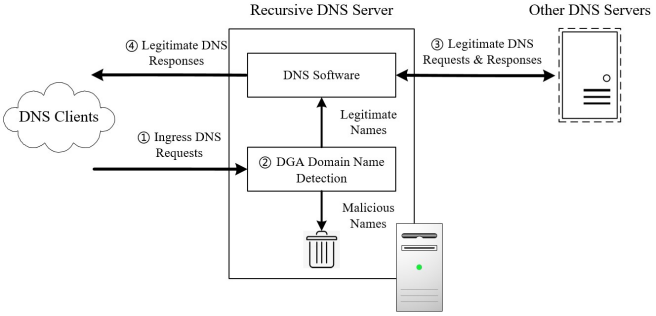

Fig. 2. DGA domain name detection process

Figure 2 depicts an overview of DGA domain name detection. the Recursive DNS Server examines incoming DNS requests originating (Step ① and Step ②). Any malicious names DNS requests are dropped, while legitimate names are resolved by the DNS Software, which is integrated within the Recursive DNS Server (Steps ③ and ④).

A. Model based on recurrent neural network

RNN (Recurrent Neural Network) performs well in extracting information from text sequence data, but gradient explosions or disappearance makes RNN unstable and difficult to capture long-term memory.

LSTM (Long Short-Term Memory) is a kind of RNN that introduces a gate mechanism to control the inflow and loss of features. By combining short-term memory with long-term memory, gradient disappearance can be solved, and LSTM is widely used in DGA domain name detection.

Woodbridge et al. [8] first introduced deep learning into DGA domain names detection. They presented a DGA classifier model that leveraged LSTM networks for real-time prediction of DGAs without the need for contextual information or manually created features. The detection process is shown in Figure 3. The classifier model achieved a good detection effect and could accurately perform multi-classification. This study laid the foundation for subsequent research.

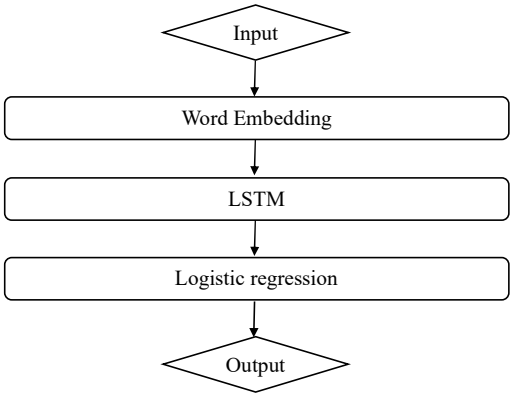

Fig. 3. Flow chart of LSTM detection method

Gated Recurrent Unit (GRU) is a kind of RNN and also a variant of LSTM. Chen et al. [9] proposed a detection model based on GRU that performed better than detection methods based on Support Vector Machine (SVM) or logical regression in various classification metrics and the model converged quickly and smoothly.

BiLSTM consists of backward and forward hidden layers to access the preceding and succeeding context of a sequence.

Shahzad et al. [10] compared the effects of different RNN models on DGA domain name detection. The BiLSTM model performed better than the LSTM model or GRU model and had a simpler structure and the shortest training time. Overall, the detection effects of the three models were similar.

In the actual network environment, the number of DGA domain names varies greatly among families. Some families in the training dataset have a very small number, which leads to the problem of sample imbalance issues.

In order to solve this problem, Tran et al. [11] introduced cost-sensitive loss function and proposed an LSTM-MI model. Chen et al. [12] proposed an LSTM.PQDO model. It took the original number and characteristics of domain names into consideration, iterated the resampling proportion of the optimal solution, and heuristically searched for a better solution around the initial solution. These two models could achieve better performance compared with existing models to overcome the difficulties of imbalanced datasets.

Niu et al. [13] used LSTM with Bayesian optimization neural network to optimize the hyperparameter combination, and the accuracy of the model reached more than 97%, which had superior performance compared with the conventional model and effectively improved the accuracy in the DGA domain names detection and classification.

B. Model based on convolutional neural network

CNN is commonly used in natural language processing, computer vision, and other fields [14]. CNN can be used to extract the relationship between local features of text data. Zhang et al. [15] proved that CNN with character-level word embedding had good performance in text classification, which provided a research direction for DGA domain name detection.

Saxe et al. [16] proposed an eXpose model that used character-level word embedding and CNN to automatically extract features. Yu et al. [17] designed a PCNN model that used three parallel CNNs to simultaneously extract the 2-gram, 3-gram, and 4-gram features of DGA domain names. The model improved the detection effect compared to the model using only one CNN.

Zhou et al. [18] designed a CNN model based on time to extract implicit features of domain names. In addition to extracting local features of domain names, temporal features were also added, thus improving the performance in DGA domain name detection.

C. Model based on hybrid neural network

In the field of text classification, hybrid neural networks have achieved remarkable results [19]. Inspired by this, some researchers have combined the advantages of CNN and RNN in the detection of DGA domain names. On the one hand, the combinatorial neural network structure can perceive the local features of the domain name, on the other hand, long-term time sequence information can be extracted.

Pei et al. [20] compared and analyzed various models and found that the DGA domain names detection model constructed by the combination of CNN and Bi-GRU had better detection ability. Experiments showed that using the CNN-LSTM model [21],[22] to extract and fuse domain name character features had a better detection effect than using CNN or LSTM alone. The recall and F1 score of the proposed models were superior to other comparative models which were solely composed of CNN or LSTM.

The model based on a hybrid neural network combined the advantages of different networks, but there were still some problems such as sample imbalance and weak model generalization ability [23].

III. Detection Method

A. Multi-dimensional features

Historically, DGA domain name detection predominantly relied on character-level features, segmenting each character individually. This method often fails to capture the underlying semantic nuances inherent in the characters that comprise domain names. Experimental results [24] validate that that a special DGA domain name, ReplaceDGA, which simulates the hidden semantic relationships within benign domain names during its generation, successfully evades various character-level DGA classifiers.

To enhance the extraction of semantic information from domain name characters, this section conducted statistical analysis on DGA domain name strings. A domain name is composed of numbers, letters, and special symbols. Once the Top-level domain (such as ".com", ".net", and ".org") is removed, the remaining part of the domain name can be broken down into its constituent characters. These characters include 38 legal characters (a-z, 0-9, "-", "."), and can be used as character features in analysis.

In the field of NLP (Natural Language Processing), the part-of-speech of English words is a commonly used data source, and many related studies revolve around the annotation of part-of-speech.

The part-of-speech can provide relevant features beyond a single character, aiding the model in the detection and classification of DGA domain names. For instance, some DGA domain names are wordlist-based, formed by randomly combining several words from a list, and their grammatical sequence characteristics differ from those of normal domain names. Specifically, a domain name like "theirkill.com" belongs to the suppobox family, and its grammatical sequence is (adjective, verb). This combination deviates from the norms of standard language usage.

McDonald et al. [25] proposed features of the N-gram of text part-of-speech sequences to classify sensitive text, and their research proved the feasibility of applying the information provided by part-of-speech sequences to solve text classification problems.

Additionally, Domain names are inherently not as lengthy as sentences. Some DGA domain names are even shorter than typical domain names, falling into the category of shorter-length DGA domain names and not being composed of recognizable words. Therefore, it is sensible to consider 2-gram features. For instance, "jvrg.org" is a domain name from the conficker family, due to its short length, the information contained in a single character is limited. By employing 2-gram features, additional information can be extracted.

To summarize, we propose nine categories of semantic features. Multi-dimensional features include character features and semantic features, as illustrated in Table II.

TABLE II
MULTI-DIMENSIONAL FEATURE

Sequence Number	Feature Type	Description
1-26	*Character feature*	*corresponding to a through z, respectively*
27-36	*Character feature*	*corresponding to 0 through 9, respectively*
37	*Character feature*	*the symbol '-'*
38	*Character feature*	*the symbol '.'*
39	*Semantic feature*	*any digit from 0 to 9*
40	*Semantic feature*	*the symbol '-'*
41	*Semantic feature*	*the symbol '.'*
42	*Semantic feature*	*any letter from a to z*
43	*Semantic feature*	*bigram class (2-gram)*
44	*Semantic feature*	*noun class (including common nouns, personal pronouns, etc.)*
45	*Semantic feature*	*verb class*
46	*Semantic feature*	*adjective class (including adjective, adverb, etc.)*
47	*Semantic feature*	*other part of speech (including conjunction, auxiliary words, modal particles, etc.)*

The 38 legal characters are traditionally classified as character features, but they also carry semantic information, which justifies their inclusion in the semantic feature set.

We differentiate by grouping digits (0-9) into a single category with a sequence number of 39, and all single letters into another distinct category with a sequence number of 42, marking a distinction from traditional character feature categorization.

In addition, semantic features encompass bigrams and English words, which can be categorized into four groups based on the part of speech: Noun, Verb, Adjective, and Other. We assign a unique sequence number to each category.

B. The structure of the detection model

To extract information more effectively for detection, this paper combines the structures of different neural networks to construct the model, as shown in Figure 4.

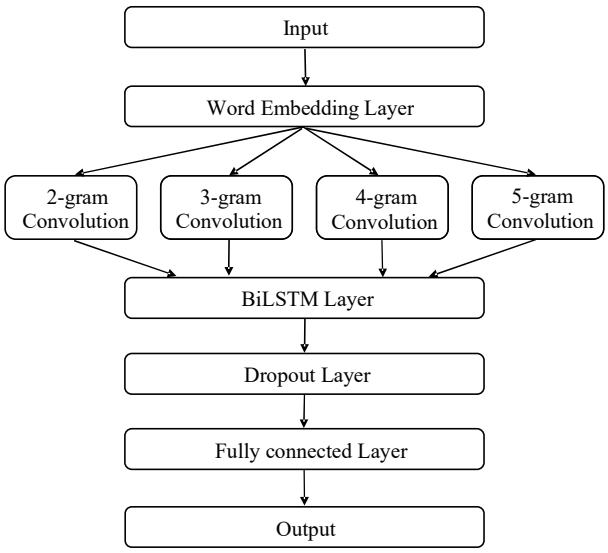

Fig. 4. The structure of detection model

The multi-dimensional features of domain names are extracted through the word embedding layer, which maps the domain names into character-level word embeddings. Subsequently, feature extraction is automatically performed in a hybrid neural network layer. This hybrid layer includes a convolutional layer, a concatenation layer, a BiLSTM layer, and a dropout layer. Ultimately, a fully connected neural network is utilized for classification.

1) Word embedding layer

Word embedding is a processing method in NLP. Since domain names are short texts, deep learning algorithms cannot directly process them. The subsequent modeling requires the input to be an array, so the domain name needs to be encoded.

Each character in each multi-dimensional feature sequence is converted to a vector $\mathbf{a}_n$ by using one-hot encoding. The matrix $\mathbf{A} \in \mathbf{R}^{m \times n}$ is obtained, which is used as the input of the word embedding layer. As shown in formula (1), where *m* is the length of the encoding dictionary.

$$\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, ..., \mathbf{a}_n) \tag{1}$$

(1) Word embedding and output

Word embedding uses neural networks. In the process of word embedding, the corresponding weight parameters $\mathbf{W} \in \mathbf{R}^{m \times L}$ will be constantly updated according to the backpropagation with the iterative training of the neural network, and the word vector of each character will be constantly optimized.

$$\mathbf{X} = \mathbf{W}^{\mathrm{T}} \cdot \mathbf{A} \tag{2}$$

The output domain word vector matrix is expressed as $\mathbf{X} \in \mathbf{R}^{L \times n}$, each domain word vector matrix has *n* word vectors. Where, $\mathbf{x}_i$ is the word vector corresponding to the i-th character in the integer sequence, with a dimension of L.

$$\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3}, ..., x_{iL})^{\mathrm{T}} \tag{3}$$

$$\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_n)^{\mathrm{T}} \tag{4}$$

(2) Construction input

After removing the Top-level domain, we construct character feature sequences and semantic feature sequences, respectively, as outlined in Table II.

WordNinja is used as a word segmentation tool, and nltk is

employed for marking each segmented part with semantic categories, acting as a labeling tool. Subsequently, we encode semantic features by category to obtain the semantic sequence.

Finally, the semantic feature encoding sequence of the domain name is directly concatenated with the character encoding sequence to form a multi-dimensional feature sequence. Additionally, any insufficient integer sequence is padded with zeros.

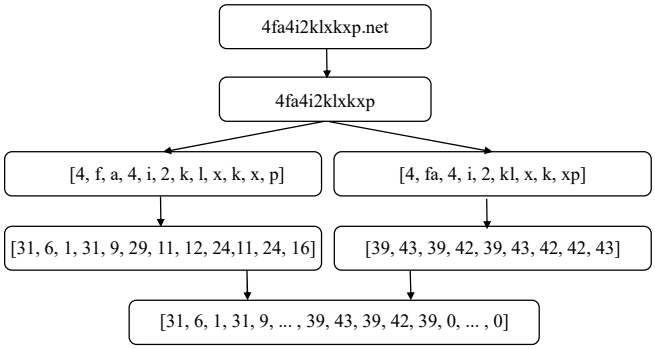

Fig. 5. The complete word embedding process

Taking the DGA domain name "4fa4i2klxkxp.net" as an example, the complete word embedding process is shown in Figure 5.

2) Hybrid neural network layer

A hybrid deep neural network, which includes CNN and BiLSTM, is used to extract and fuse features of DGA domain names.

(1) Convolutional layer

In the convolutional layer, this paper employs four parallel convolutional neural networks that simultaneously receive input from the word embedding layer and extract different local features.

In various convolutional neural networks, the height of the convolutional kernel is represented by **k**, and **s** represents the convolutional kernel. In the domain word vector matrix **X**, the convolutional kernel slides over the window vector $\mathbf{w}_j$ to obtain a feature map **z**, the calculation formula is (5), where *f* is the nonlinear activation function, *b* is the bias term and *n* is the length of the feature sequence.

$$\mathbf{z} = f\left(\sum_{j=1}^{n-k+1}(\mathbf{w}_j \cdot \mathbf{s}) + b\right) \tag{5}$$

The feature map $\mathbf{z} \in \mathbf{R}^{n-k+1}$ obtained by each convolutional kernel can be represented as (6).

$$\mathbf{z} = (z_1, z_2, ..., z_{n-k+1})^{\mathrm{T}} \tag{6}$$

Finally, the feature maps obtained by each convolutional kernel are concatenated column-wise to obtain the overall feature map **Z** of the convolutional neural network.

$$\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_{n\text{-}k+1}) \tag{7}$$

(2) Concatenate layer

Generally, the pooling layer is set after convolution for feature dimension reduction, so that features extracted by a convolutional neural network can be more significant. However, the original features may be lost and the dependency between domain name characters may be destroyed.

Therefore, after the convolutional layer, a feature fusion layer is designed, and features are spliced and fused by vector join operation to obtain a one-dimensional feature fusion vector.

(3) BiLSTM layer

BiLSTM is a variant of RNN models and has been widely used in text analysis. BiLSTM consists of both backward and forward hidden layers, allowing it to access the preceding and succeeding context of the sequence.

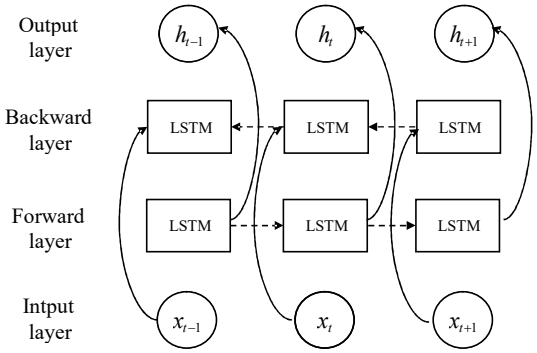

Fig. 6. The structure of BiLSTM

There are two independent LSTMs in BiLSTM. $\mathbf{x}_t$ is the input, and $\mathbf{h}_t$ is based on the outputs of forward LSTM and backward LSTM. The structure of forward LSTM and backward LSTM are the same.

When new information is added, some old information needs to be forgotten through forget gate f.

The output of $\mathbf{f}_t$ is between 0 and 1, where 1 means "completely keep" and 0 means "completely discard".

$$\mathbf{f}_t = \sigma(\mathbf{W}_\mathbf{f} \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_\mathbf{f}) \tag{8}$$

$\mathbf{i}_t$ decides what information needs to be updated, and $\tilde{\mathbf{c}}_t$ is the potential updated content.

$$\mathbf{i}_t = \sigma(\mathbf{W}_\mathbf{i} \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_\mathbf{i}) \tag{9}$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_\mathbf{c} \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_{\tilde{\mathbf{c}}}) \tag{10}$$

$\mathbf{i}_t$ it and $\tilde{\mathbf{c}}_t$ are used to add new information to the current state. Then $\mathbf{c}_t$ could be updated as follows:

$$\mathbf{c}_t = \mathbf{f}_t \cdot \mathbf{c}_{t-1} + \mathbf{i}_t \cdot \tilde{\mathbf{c}}_t \tag{11}$$

A sigmoid function is applied to determine which part will be the output $\mathbf{o}_t$. Finally, getting the output $\mathbf{h}_t$ as represented in equation (13):

$$\mathbf{o}_t = \sigma(\mathbf{W}_\mathbf{o} \cdot (\mathbf{h}_{t-1}, \mathbf{x}_t) + \mathbf{b}_\mathbf{o}) \tag{12}$$

$$\mathbf{h}_t = \mathbf{o}_t \cdot \tanh(\mathbf{c}_t) \tag{13}$$

In the BiLSTM structure, the input in both directions are processed independently. The two output vectors are concatenated as the final feature output, as represented in equation (14):

$$\mathbf{h} = (\overleftarrow{\mathbf{h}}, \overrightarrow{\mathbf{h}}) \tag{14}$$

(4) Dropout layer

Since the entire model is a composite model, its structure becomes overly complex during training. Therefore, we

employ Dropout regularization to reduce the number of neurons and enhance the model's generalizability.

(5) Fully connected layer

To facilitate subsequent classification, one dense layer is used to perform nonlinear transformations on the features, followed by another dense layer for the final classification.

Finally, the model uses a loss function for training based on backward propagation.

IV. Experiment Design

A. Dataset construction

The datasets for our experimental studies were divided into two categories: DGA datasets and normal datasets.

Normal datasets came from the top one million data released by Qi'an Xin company. Qi'an Xin has studied various considerations for domain name ranking designs. An empirical evaluation of the generated domain name ranking demonstrates that it has better stability and resistance to manipulation than existing domain name rankings, such as Alexa and Tranco [26].

As shown in Table III, DGA datasets included a total of 59 DGA families, which were from 360 netlab. Based on the proportion of the number of DGA families, 250,000 DGA domains from a total of one million DGA domains were selected to obtain the final DGA dataset.

To ensure a balanced ratio between positive and negative samples, we randomly sampled 25,000 normal domain names to constitute the normal dataset.

A total of 500,000 data pieces were constructed into the complete dataset. Finally, the dataset was divided into a training set, a validation set, and a test set in a ratio of 48:1:1.

TABLE III
THE DGA DATASET

DGA family	Tag	Data count	DGA family	Tag	Data count
wauchos	1	1359	blackhole	31	22
virut	2	11965	ccleaner	32	11
tinba	3	23193	chinad	33	1000
tempedreve	4	600	conficker	34	498
symmi	5	965	copperstealer	35	18
suppobox	6	2819	cryptolocker	36	1000
simda	7	6844	dmsniff	37	133
shiotob	8	1820	dyre	38	1000
shifu	9	579	enviserv	39	492
rovnix	10	40749	feodo	40	263
ranbyus	11	3066	fobber_v1	41	297
ramnit	12	4587	fobber_v2	42	298
qakbot	13	1128	gspy	43	100
qadars	14	458	kfos	44	121
pykspa_v1	15	10074	m0yv	45	69
ngioweb	16	1200	madmax	46	16
necurs	17	1840	matsnu	47	906
necro	18	661	nymaim	48	479
mydoom	19	2287	omexo	49	40
murofet	20	1943	padcrypt	50	168
monerominer	21	576	proslikefan	51	100
locky	22	260	pykspa_v2_fake	52	799
gameover	23	2709	pykspa_v2_real	53	198
flubot	24	6777	tinynuke	54	32
emotet	25	1351	tofsee	55	20
banjori	26	109492	tordwm	56	500
abcbot	27	27	vawtrak	57	844
antavmu	28	32	vidro	58	100
bamital	29	104	xshellghost	59	10
bigviktor	30	1000	**Total**	250000	

TABLE IV
EXPERIMENTAL GROUP

Experimental group	Using character feature	Using multi-dimensional feature	Deep neural network
CNN	✓	-	CNN
CNN-M	-	✓	CNN
BiLSTM	✓	-	BiLSTM
BiLSTM-M	-	✓	BiLSTM
CNN-BiLSTM	✓	-	CNN and BiLSTM
CNN-BiLSTM-M	-	✓	CNN and BiLSTM

B. Experiment and parameter settings

We designed three deep neural network models and conducted experiments by combining different word embedding methods for each model, in order to perform a comparative analysis.

As shown in Table IV, six experimental groups were designed to compare the performance in detecting DGA domain names.

The experimental parameters were set as follows:

1) Word embedding layer

According to the statistical analysis of all domain names, the maximum length of the character feature sequence for domain names was 75, and the maximum length of the multi-dimensional feature sequence was 100, so the standard lengths were set to 75 and 100 respectively for the character feature sequence and the multi-dimensional feature sequence.

Word Embedding was applied to obtain the corresponding word vector, and the dimension of the word embedding was generally set to 32.

2) *Hybrid neural network layer*

For the parallel CNNs, the sizes of the convolution kernels, k, were 2, 3, 4, and 5, respectively. And *m*, which was the number of kernels, was set to 256. ReLU was selected as the activation function for the CNN.

For BiLSTM, the parameters and input matrices were consistent, and the number of neurons in BiLSTM was set to 256. The parameter for Dropout regularization was set to 0.3.

3) Fully Connected layer

The fully connected layer was responsible for completing the final classification and outputting the results. The cross-entropy loss function was utilized, and Adam optimization algorithm was employed to accomplish the backward propagation and train the model.

In two-classification, the sigmoid function served as the activation function. In multi-classification, the softmax function was utilized as the activation function.

V. ANALYSIS OF RESULTS

A. Two-classification results analysis

In two-classification, DGA domain names, which were the positive samples, were marked by 1. Normal domain names, which were the negative samples, were marked by 0. Figures 7 and 8 display the ROC curves of each model.

We focused on the AUC, the area under the ROC curve. Comparing Figure 7 with Figure 8, models that used multi-dimensional features had higher AUC values than models that used only character features. The small AUC gap between models was significant in application.

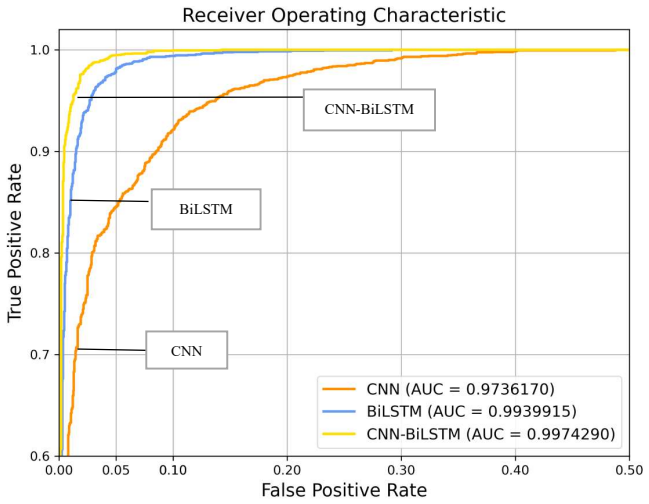

Fig. 7. The ROC curves of the model using character features

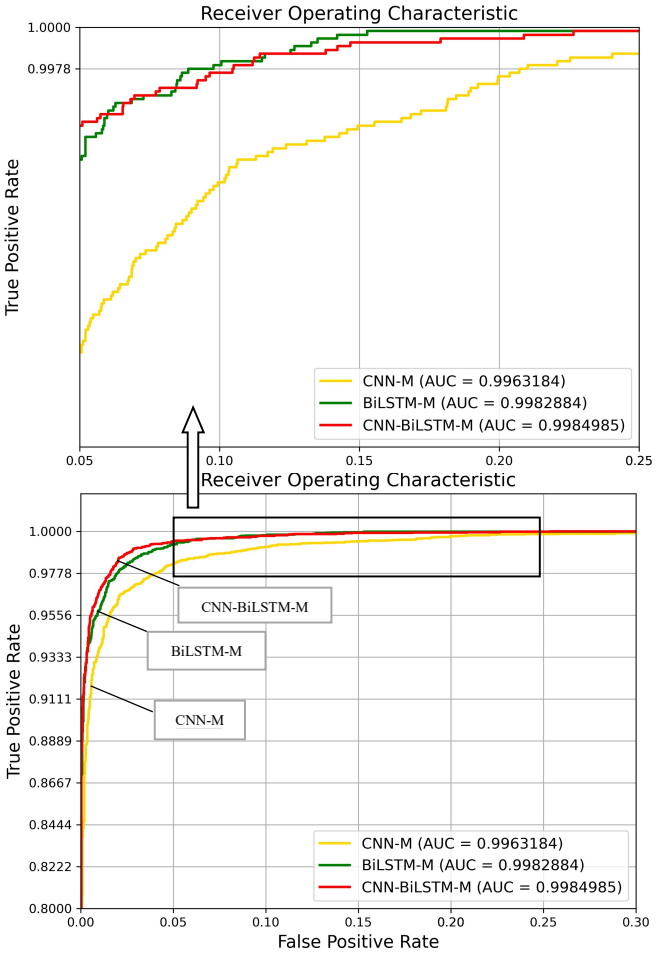

Fig. 8. The ROC curves of the model using multi-dimensional features

The CNN-BiLSTM-M model provided the best performance, with an AUC of 0.9984985, as shown in Figure 8. This means that the model could be capable of accurately identifying DGA domain names, showing high classification accuracy and robustness.

In two-classification, precision, recall, and F1 score were evaluation metrics. The results are shown in Figure 9. It could be seen that the detection performance of the three models using multi-dimensional features in word embedding was significantly improved.

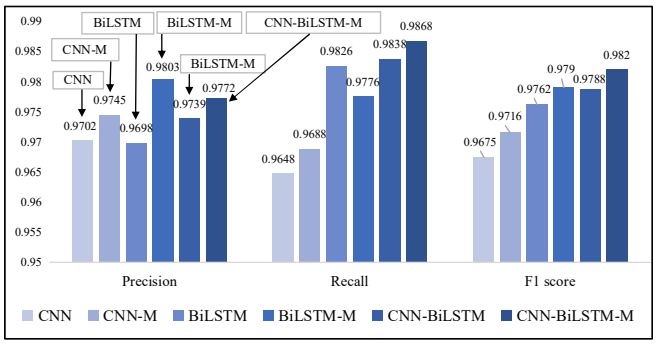

Fig. 9. Two-classification results

Three evaluation metrics for all models were above 96%. Compared with the CNN model, all evaluation metrics of the CNN-M model increased by about 0.4%. Compared with the BiLSTM model, the precision and F1 score of the BiLSTM-M model also improved by about 0.2%. Compared with the CNN-BiLSTM model, all evaluation metrics of the CNN-BiLSTM-M model increased by about 0.3%.

As the number of different DGA families was unbalanced, the F1 score could well reflect the comprehensive performance of the model. The CNN-BiLSTM-M model had the highest F1 score in DGA domain name detection, which was 98.20%.

In summary, in the two-classification of DGA domain names, results indicated that multi-dimensional feature embeddings performed well and could enhance the extraction of domain name features, thereby improving the effectiveness of the deep learning model.

The CNN-BiLSTM-M model further strengthened the ability to extract and fuse multi-dimensional features and improved the accuracy of classification as well as the model's generalization ability.

As shown in Table V, we compared the two-classification results between the current mainstream models and the proposed model in this paper.

TABLE V
COMPARISON OF TWO-CLASSIFICATION RESULTS FOR DIFFERENT METHODS

Experimental group	Precision	Recall	F1 score
AN+LSTM	0.9550	0.9428	0.9495
CNN+BiGRU	0.9438	0.9325	0.9385
LSTM+Attention	0.9502	0.9491	0.9504
CNN-BiLSTM-M	0.9772	0.9868	0.9820

The comparative models included the AN+LSTM model [27], CNN+BiGRU model [28], and LSTM+Attention model [29]. Despite the diverse nature of the domain datasets leveraged by these models, they were all meticulously designed to accurately reflect the characteristics of both DGA domain names and normal domain names. Upon thorough comparison and analysis of their performances, the CNN-BiLSTM-M model emerged as the top performer across all evaluated metrics.

The reasons were as follows:

(1) The AN+LSTM model and LSTM+Attention model only utilized word embeddings composed of the characters in domain names, lacking the extraction of other semantic features within the domain names. Furthermore, both models primarily employed the LSTM model and incorporated the Attention mechanism to rank the importance of the correlations between the LSTM inputs and outputs. This method took into account the weights of different characters in various positions within the DGA domain name and achieved higher classification accuracy compared to the simple LSTM algorithm. However, it lacked the extraction of local features such as n-grams from the domain name text, further resulting in suboptimal detection performance.

(2) The CNN+BiGRU model utilized a fused word embedding that combined domain name characters and radix combinations. However, this method was highly targeted, demonstrating good detection accuracy primarily for three DGA domain name families generated based on dictionaries: matsnu, suppobox, and ngioweb. It did not show significant effects on other DGA domain name families.

(3) The CNN-BiLSTM-M model proposed incorporated semantic features. It utilized parallel CNNs to extract local N-gram features from domain name characters and employed Bi-LSTM to capture the dependencies between domain name features, thereby reducing the loss of domain name feature information. This enhancement led to improved detection performance for most DGA domain name families. The overall performance and detection accuracy of the model were further enhanced.

B. Multi-classification results analysis

In multi-classification, DGA domain names were marked according to their DGA family, with values ranging from 1 to 59. Normal domain names were marked with 0.

1) Model performance evaluation

The values of precision, recall, and F1 score in weighted average were used for evaluation in the multi-classification of DGA domain names.

As shown in Figure 10, the evaluation metrics for the CNN model, BiLSTM model, and CNN-BiLSTM model had improved, indicating that multi-dimensional features could further enhance the deep learning model's ability to extract the implicit semantic features of DGA domain names.

Under the weighted average, the three metrics of the CNN-M model showed a slight improvement compared with the CNN model, about 1%. The F1 score of the CNN-M model was improved in 26 DGA families.

The three metrics of the BiLSTM-M model under the weighted average showed some improvement compared with the BiLSTM model, and the F1 score showed the most improvement, increasing by 0.19%.

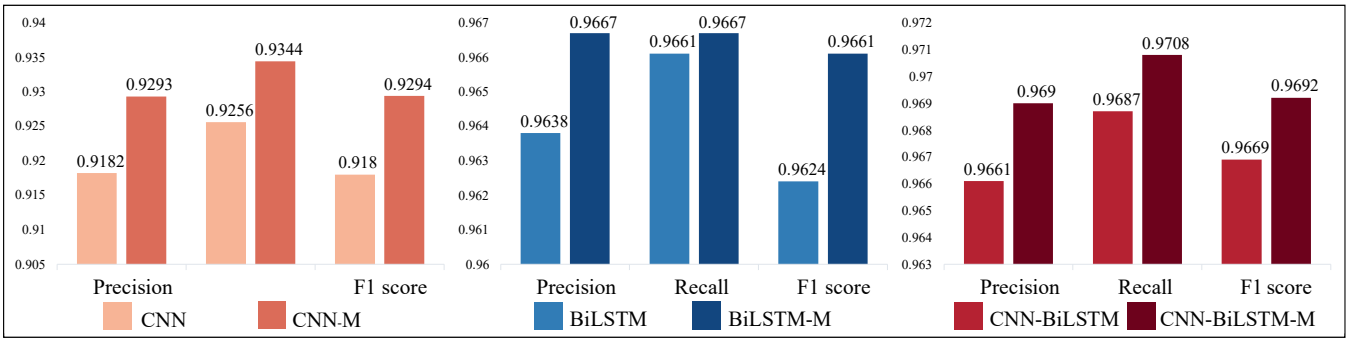

Fig. 10. Comparison of different word embedding models

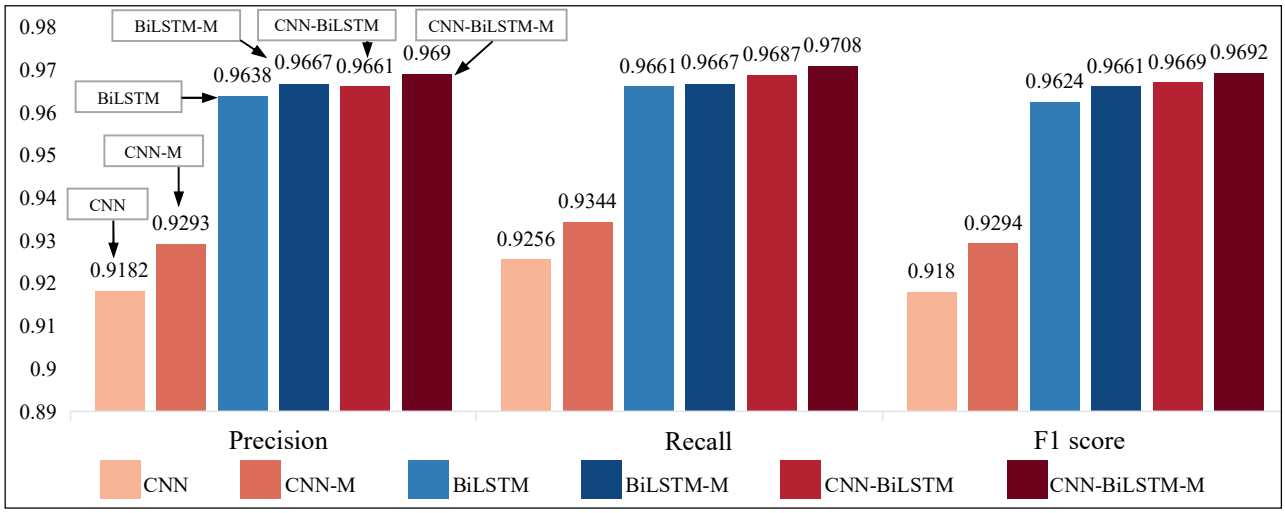

Fig. 11. The comparison of all experimental groups

In terms of detection effectiveness for each DGA family, the F1 score of the BiLSTM-M model has improved to varying degrees compared to the BiLSTM model across 35 DGA families.

The three metrics of the CNN-BiLSTM-M model, when calculated using a weighted average, showed significant improvement compared to those of the CNN-BiLSTM model, with an increase of approximately 0.2%. In terms of detection ability for each DGA family, the F1 score of the CNN-BiLSTM-M model improved to varying degrees compared to that of the CNN-BiLSTM model, across 32 DGA families.

Comparing all the results shown in Figure 11, it became clear that the CNN-BiLSTM-M model achieved the pinnacle in evaluation metrics, with a precision rate of 96.90% and an F1 score of 96.92% for DGA domain name detection. The performance indicated that it had the best detection ability.

The results aligned with theoretical expectations, demonstrating that the employment of multi-dimensional feature word embedding, enriched with semantic features, significantly augments the capacity to discern and extract the latent semantic characteristics inherent in DGA domain names.

2) The detection effectiveness of DGA Families

In order to further compare and analyze the detection effectiveness of various models on DGA domain families, we focused on two specific families, which were crucial in evaluating the overall performance of the model. Zero-detection families had 0% on all evaluation metrics, while full-detection families scored 100% on all evaluation metrics.

As presented in Table VI, firstly, we analyzed the effect of multi-dimensional features. Compared to the CNN model, the number of zero-detection families decreased and the number of full-detection families increased in the CNN-M model. Similarly, when comparing the BiLSTM-M model with the BiLSTM model, and the CNN-BiLSTM-M model with the CNN-BiLSTM model, we observed a general reduction in the number of zero-detection families and an increase in the number of full-detection families.

TABLE VI
STATISTICS OF ZERO-DETECTION FAMILIES AND FULL-DETECTION FAMILIES

Experimental group	Number of zero-detection families	Number of full-detection families
CNN	15	1
CNN-M	13	1
BiLSTM	4	8
BiLSTM-M	3	9
CNN-BiLSTM	6	10
CNN-BiLSTM-M	4	11

Then, the CNN-BiLSTM-M model exhibited a remarkably low number of zero-detection families, with only four DGA

domain families. Simultaneously, it boasted the highest count of full-detection families, successfully identifying 11 DGA domain families.

This also explained why the CNN-BiLSTM-M model could achieve the best performance in multi-classification compared to other comparative models.

Furthermore, a detailed analysis was conducted on the detection effectiveness of the CNN-BiLSTM-M model for various DGA families. As depicted in Figure 12, using Sturges' Formula from statistics, we calculated that the optimal number of groups is approximately six. Therefore, we divided the F1 scores of all DGA domain families into six categories.

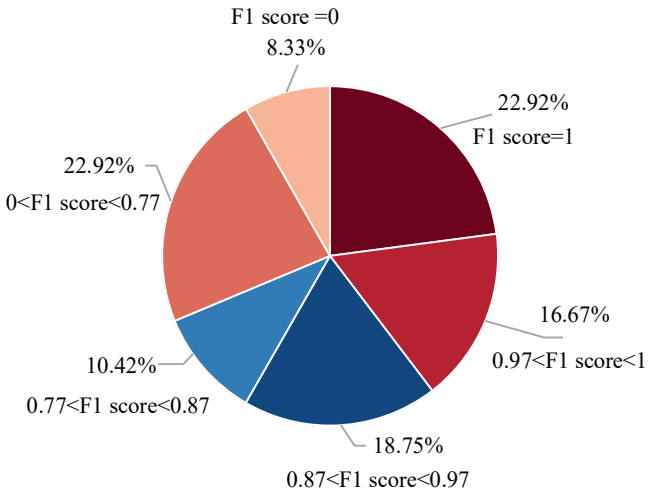

Fig. 12. The distribution of F1 score

The mean of the F1 score for all detected DGA families was 0.77, while the F1 score in weighted average was 0.9692, approximately equal to 0.97. The weighted average F1 score was higher. According to Figure 12, DGA families with the F1 score greater than 0.97 accounted for approximately 40% of the total. In reality, these families also comprised a larger proportion of domain names in the dataset, resulting in an overall excellent detection performance of the model.

TABLE VII
DGA FAMILIES WITH F1 SCORES BELOW 0.77

DGA Family	Precision	Recall	F1 score
abcbot	0.0000	0.0000	0.0000
conficker	0.0000	0.0000	0.0000
proslikefan	0.0000	0.0000	0.0000
pykspa_v2_real	0.0000	0.0000	0.0000
nymaim	0.3333	0.125	0.1818
locky	0.5	0.1429	0.2222
pykspa_v2_fake	0.2353	0.3636	0.2857
qakbot	0.6111	0.4231	0.5
cryptolocker	0.7273	0.381	0.5
fobber_v1	1	0.3333	0.5
fobber_v2	0.3333	1	0.5
tempedreve	0.5333	0.8	0.64
bamital	1	0.5	0.6667
matsnu	0.7895	0.6818	0.7317
locky	0.7935	0.7449	0.7684

Table VII presented 15 DGA families with the F1 score below the mean, accounting for approximately one-third of all DGA families. Then we analyzed the reasons for the poor detection effectiveness of them.

First, apart from the locky family, the number of other DGA families in the test set was less than 0.5%, categorizing them as small sample domains. The issue of class imbalance caused by the extremely small sample size leads the model to neglect the domains of these families during training, resulting in poor recognition and detection capabilities for them.

Then, as illustrated in Figure 13, a statistical analysis was conducted on the proportion of nine semantic features across DGA families with F1 scores below 0.77 and those with F1 scores above 0.97. The comparison revealed differing shapes in the distribution of semantic features between these two types of DGA families.

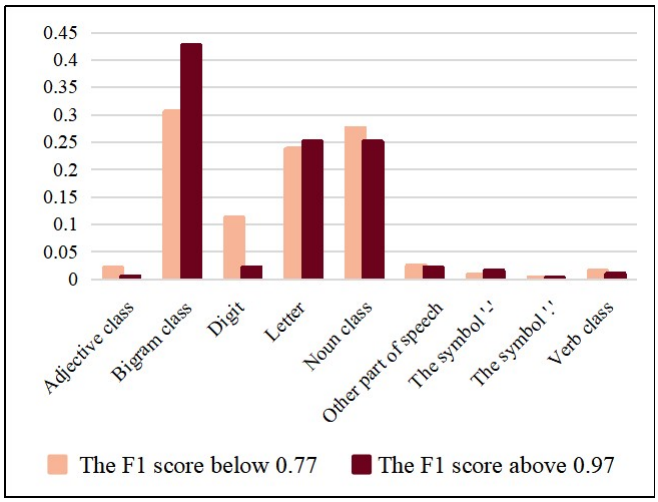

Fig. 13. Comparison of semantic features distribution

The most significant differences we found lay in the proportions of two features: the Bigram class semantic feature and the Digit class. Specifically, DGA families with F1 scores below 0.77, the proportion of the Bigram class was approximately 30%, whereas in DGA families with F1 scores above 0.97, it rose above 42%. Conversely, the proportion of the Digit class in DGA families with F1 scores above 0.97 were notably lower compared to that in DGA families with F1 scores below 0.77. The distribution of these two types of semantic features significantly influenced the effectiveness of detection.

Additionally, we observed that the length of the semantic feature sequence of DGA families with F1 scores above 0.97 domain families ranged from 1 to 22, while of DGA families with F1 scores below 0.77, it was between 3 and 16. Due to the shorter lengths of these semantic feature sequences, they contained less valid information.

Especially for the four types of zero-detection families, their semantic sequences were only 2 to 6 in length, which was extremely short, and they only possessed four types of semantic features: Digit class, Bigram class, Noun class, and Verb class. This resulted in their inability to be detected.

A closer examination of the 11 full-detection families revealed intriguing consistencies. For instance, the omexo family consistently utilized fixed-length hash hexadecimal, each with a distinct length of 32 characters. Conversely, the

symmi family adopted a unique approach by employing fixed combinations of letters, specifically a blend of vowels and consonants.

This underscored the proficiency of the CNN-BiLSTM-M model in extracting latent character length information and semantic nuances from these types of DGA families, resulting in exceptional detection performance for DGA domain names that were generated with fixed lengths or fixed letter patterns.

VI. Conclusion

Detecting DGA domain names poses a major challenge in the field of cyber security. In this paper, we addressed some of the existing problems in DGA domain name detection by refining the word embedding method and designing a hybrid deep neural network model.

We propose an effective model known as CNN-BiLSTM-M. This hybrid deep neural network detection model combines the advantages of CNN and BiLSTM, capturing both local features and long-distance dependencies within domain name sequences. It also avoids the disadvantages associated with single neural network structures in domain name feature extraction.

The experimental results showed that, compared to other models, the CNN-BiLSTM-M model, equipped with multi-dimensional features, was able to extract significantly more information from domain names. This led to a marked improvement in detection performance for the majority of DGA families, demonstrating the model's superior detection accuracy and generalization capabilities.

In summary, the detection method presented in this paper has demonstrated effectiveness in both two-classification and multi-classification. Further research on the following aspects is warranted.

There are some DGA families with small sample sizes in the experimental dataset, which made the detection ineffective or even failed to detect at all. The issue of how to effectively detect DGA domain names with small samples deserves further research.

Additionally, when constructing the input for word embedding, the semantic feature sequence is directly concatenated with the character sequence. However, the semantic and character features of domain names are fundamentally different types of data. Therefore, this concatenation method might result in the extraction of mixed features by the model. In the future, further research could be conducted on feature integration techniques, and alternative features could be proposed.

References

[1] Youfeng Niu, Mingxi Guan, Wenhao Yuan, Yilin Chen, Lingyi Chen, Qiming Yu, "A Bayesian optimization-based LSTM Model for DGA Domain Name Identification Approach," Journal of Physics: Conference Series: 2022 2nd International Conference on Artificial Intelligence and Industrial Technology Applications, 13-15 May, 2022, Dali, China, pp012015.

[2] Jafari Dehkordi Mohammad and Sadeghiyan Babak, "Reconstruction of C&C Channel for P2P Botnet," IET Communications, vol.14, no.8, pp1318-1326, 2020.

[3] Nikos Kostopoulos, Dimitris Kalogeras, Dimitris Pantazatos, Maria Grammatikou and Vasilis Maglaris, "SHAP Interpretations of Tree and Neural Network DNS Classifiers for Analyzing DGA Family Characteristics," IEEE Access, vol.11, pp61144-61160, 2023.

[4] Futai Zou, Yue Tan, Lin Wang, Yongkang Jiang, "Botnet Detection Based on Generative Adversarial Network," Journal on Communications, vol.42, no.7, pp95-106, 2021.

[5] Daniel Plohmann, Khaled Yakdan, Michael Klatt, Johannes Bader, and Elmar Gerhards-Padilla, "A Comprehensive Measurement Study of Domain Generating Malware," USENIX Association: Proceedings of the 25th USENIX Conference on Security Symposium, 10–12 August, 2016, Austin, USA, pp263–278.

[6] Ahmad O. Almashhadani, Mustafa Kaiiali, Domhnall Carlin, Sakir Sezer, "MaldomDetector: A System for Detecting Algorithmically Generated Domain Names with Machine Learning," Computers & Security, vol.93, pp101787, 2020.

[7] Haofan Wang, "Botnet Detection via Machine Learning Techniques," IEEE: 2022 International Conference on Big Data, Information and Computer Network, 20-22 January, 2022, Sanya, China, pp831-836.

[8] Jonathan Woodbridge, Hyrum S. Anderson, Anjum Ahuja, Daniel Grant, "Predicting Domain Generation Algorithms with Long Short-Term Memory Networks," arXiv Preprint arXiv: 1611.00791, 2016.

[9] Liguo Chen, Yuedong Zhang, Guanggang geng, Zhiwei Yan, "Detection of Random Generated Names Using Recurrent Neural Network with Gated Recurrent Unit," Computer Systems and Applications, vol.27, no.8, pp198-202, 2018.

[10] Haleh Shahzad, Abdul Rahman Sattar and Janahan Skandaraniyam, "DGA Domain Detection using Deep Learning," IEEE: 2021 IEEE 5th International Conference on Cryptography, Security and Privacy ,08-10 January, 2021, Zhuhai, China, pp139-143.

[11] Duc Tran, Hieu Mac, Van Tong, Hai Anh Tran, Linh Giang Nguyen, "A LSTM Based Framework for Handling Multiclass Imbalance in DGA Botnet Detection," Neurocomputing, vol.75, pp2401-2413 2018.

[12] Yijing Chen, Bo Pang, Guolin Shao, Guozhu Wen and Xingshu Chen, "DGA-based Botnet Detection toward Imbalanced Multiclass Learning," Tsinghua Science and Technology, vol.26, no.4, pp387-402, 2021.

[13] Weina Niu, Tianyu Jiang, Xiaosong Zhang, Jiao Xie, Junzhe Zhang, Zhenfei Zhao, "Fast-flux Botnet Detection Method Based on Spatiotemporal Feature of Network Traffic," Journal of Electronics & Information Technology, vol.42, no.8, pp1872-1880, 2020.

[14] Yoon Kim, "Convolutional Neural Networks for Sentence Classification," Association for Computational Linguistics: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 25-29 October, 2014, Doha, Qatar, pp1746-1751.

[15] Xiang Zhang, Junbo Zhao, and Yann LeCun, "Character-level Convolutional Networks for Text Classification," MIT Press: Proceedings of the 28th International Conference on Neural Information Processing Systems, 7-12 December, 2014, Cambridge, MA, USA, pp649-657.

[16] Joshua Saxe, Konatantin Berlin, "eXpose: A Character-Level Convolutional Neural Network with Embeddings For Detecting Malicious URLs, File Paths and Registry Keys," arXiv Preprint ArXiv: 1702.08568, 2017.

[17] Bin Yu, Jie Pan, Jiaming Hu, Anderson Nascimento and Martine De Cock, "Character Level based Detection of DGA Domain Names," IEEE: 2018 International Joint Conference on Neural Networks, 08-13 July, 2018, Rio de Janeiro, Brazil, pp1-8.

[18] Shaofang Zhou, Lanfen Lin, Junkun Yuan, Feng Wang, Zhaoting Ling, and Jia Cui, "CNN-based DGA Detection with High Coverage," IEEE: 2019 IEEE International Conference on Intelligence and Security Informatics, 01-03 July, 2019, Shenzhen, China, pp62–67.

[19] Doha Taha Nor El-Deen, Rania Salah El-Sayed, Ali Mohamed Hussein, and Mervat S. Zaki, "Multi-label Classification for Sentiment Analysis Using CBGA Hybrid Deep Learning Model," Engineering Letters, vol.32, no.2, pp340-349, 2024.

[20] Lanzhen Pei, Yingjun Zhao, Zhe Wang, Yunqian Luo, "Comparison of DGA Domain Detection Models Using Deep Learning," Computer Science, vol.46, no.5, pp111-115, 2019.

[21] Bin Zhang, Renjie Liao, "Malicious Domain Name Detection Model Based on CNN and LSTM," Journal of Electronics & Information Technology, vol.43, no.10, pp2944-2951, 2021.

[22] Guotian Xu, Zhenwei Shen, "DGA Malicious Domain Name Detection Method Based on Fusion of CNN and LSTM," Netinfo Security, vol.21, no.10, pp41-47, 2021.

[23] Yu Wang, Zuchao Wang, Rui Pan, "Survey of DGA Domain Name Detection Based on Character Feature," Computer Science, vol.50, no.8, pp251-259, 2023.

[24] Xiaoyan Hu, Hao Chen, Miao Li, Guang Cheng, Ruidong Li, Hua Wu, Yali Yuan, "ReplaceDGA: BiLSTM-Based Adversarial DGA With High Anti-Detection Ability," IEEE Transactions on Information Forensics and Security, vol.18, pp4406-4421, 2023.

[25] Graham McDonald, Craig Macdonald, and Iadh Ounis, "Using Part-of-Speech N-grams for Sensitive-Text Classification," Association for Computing Machinery: Proceedings of the 2015 International Conference on The Theory of Information Retrieval, 27-30 September, 2015, New York, USA, pp381-384.
[26] Xie, Qinge, Shujun Tang, Xiaofeng Zheng, Qingran Lin, Baojun Liu, Haixin Duan and Frank Li, "Building an Open, Robust, and Stable Voting-Based Domain Top List," Usenix Association: 31st USENIX Security Symposium, 10-12 August, 2022, Boston, MA, US, pp625-642.
[27] Kang Zhou, Liang Wang, Hongwei Ding, "Detection of Malicious Domain Names Based on AN and LSTM," Computer Engineering and Applications, vol.56, no.4, pp92-98, 2020.
[28] Xiaodong Li, Yuqiang Li, Yuanfeng Song and Mengshu Hou, "New DGA Domain Name Detection Method Based on Fusion Vector," Application Research of Computers, vol.39, no.6, pp1834-1837,1844, 2022.
[29] Yanchen Qiao, Bin Zhang, Weizhe Zhang, Arun Kumar Sangaiah, and Hualong Wu, "DGA Domain Name Classification Method Based on Long Short-Term Memory with Attention Mechanism," Applied Sciences vol.9, no.20, pp4205-4214, 2019.

Rui Pan was born in 1988 and is a senior engineer at China Academy of Information and Communications Technology. His main research interests include cyber security, data governance, and data security.

Yu Wang was born in 1996 and graduated from China University of Geosciences (Beijing). Her main research interests are data mining and artificial intelligence applications.

Zuchao Wang was born in 1964, is a professor at the China University of Geosciences (Beijing). His research interests include computational intelligence methods based on evolutionary algorithms and neural networks, algorithm analysis, and design.