Integrating an Attention Mechanism and Deep Neural Network for Detection of DGA Domain Names

Fangli Ren
Institute of Information Engineering,
Chinese Academy of Sciences.
School of Cyber Security,
University of Chinese Academy of Sciences.
Email: renfangli@iie.ac.cn

Zhengwei Jiang
Institute of Information Engineering,
Chinese Academy of Sciences.
Email: jiangzhengwei@iie.ac.cn

Jian Liu
Institute of Information Engineering,
Chinese Academy of Sciences.
Email: liujian6@iie.ac.cn

***Abstract*—Domain generation algorithms (DGA) are employed by malware to generate domain names as a common practice, with which to confirm rendezvous points to their command-and-control (C2) servers. The detection of DGA domain names is one of the important technologies for command and control communication detection. Considering the randomness of the DGA domain names, recent work in DGA detection employed machine learning methods based on features extracting and deep learning architectures to classify domain names. However, these methods perform poorly on wordlist-based DGA families, which generate domain names by randomly concatenating dictionary words. In this paper, we proposed the ATT-CNN-BiLSTM model to detect and classify DGA domain names. Firstly, the Convolutional Neural Network (CNN) and bidirectional Long Short-Term Memory (BiLSTM) neural network layer was used to extract the features of the domain sequences information; secondly, the attention layer was used to allocate the corresponding weight of the extracted domain deep information. Finally, the domain feature messages of different weights were put into the output layer to complete the tasks of detection and classification. The experiment results demonstrate the effectiveness of the proposed model both on regular DGA domain names and wordlist-based ones. To be precise, we got a F1 score of 98.92% for the detection and macro average F1 score of 81% for the classification task of DGA domain names.**

Keywords*—*cybersecurity, domain generation algorithm, attention mechanism, deep learning

I. INTRODUCTION

The domain name system (DNS) is an important infrastructure of the Internet, which maps easy-to-remember hostnames to boring, hard-to-remember IP addresses. It provides critical support services for the normal operation of various domain-based Web applications, email, and distributed systems. Due to DNS traffic's ability to penetrate through the firewall [1], attackers begin to use the DNS to conduct different kinds of cyber-attacks. Some large attack organizations can even use the unique ability of DNS to construct attack behavior, which has a serious impact on the overall running of the network. In recent years, a variety of malware outbreaks have caused significant losses to the government, energy, manufacturing, and other key information infrastructure. The malware defenses is critical in cyberspace security.

Modern malware such as botnets, ransomware, and Advanced Persistent Threats often use DNS service to communicate with the command and control (C&C) server for file transfer and software updates. In order to do this, the malware must know where the C&C server to connect to. In earlier days, simple approaches might hardcode an IP or a domain name. But these methods are easy to intercept by using the blacklist, the traffic to a specific IP address can be trivially blocked, and domain names can be easily seized or sinkhole. In order to improve the reliability and robustness of the communication between C&C server and malware, malware typically takes advantage of Domain Generation Algorithms (DGA) to automatically create a large set of pseudo-random domain names , and then choose one or more effective domain names to resolve the IP address, so as to realize the communication with C2 server and avoid the blocking method of blacklist. The task of confirming whether or not a domain name is generated by a DGA is a crucial step of malware defenses.

The research on identifying DGA domains has been widely pursued in recent years. From the techniques relying on manual feature engineering [2] based on machine learning to the application of deep learning. Woodbridge [3] used a simple character-level long short-term memory networks (LSTM) model to identify DGA domains, which had a high level of effectiveness except for DGA classes that resemble English words. Tran [4] proposed a novel LSTM-based model in order to detect botnets that employ DGA to generate a large number of domains as a command and control server. Saxe [5] proposed a model based on CNN to detect DGA domain names. But they did not test the performance of the model on the wordlist-based DGA, which is a kind of DGA that simulate the composition and naming methods of normal domain names, also makes the DGA domain names more concealment and the detection more difficult. There still has not been much research on detecting DGA families made from English wordlists, which making the problem of detecting wordlist-based DGA domains more challenging.

In this paper, we proposed a deep neural network model with an attention mechanism (ATT-CNN-BiLSTM) for detection and classification of DGA domain names. The main thought behind our ensemble model is that the validity of the context inherent in domains could contain sufficient information with which to distinguish DGA domain names especially the wordlist-based ones.

In summary, contributions of this paper include the following:

- The domain names detection task is similar to the natural language processing task. We build a model to learn the semantic relationships between the domain names. CNN and BiLSTM can obtain information from past and future states. Since this method can capture the text context information more completely, we apply it in the DGA domain name detection and classification.

- By integrating attention mechanism which is capable of catching the critical information into our model, we calculate the importance of the correlation between the outputs of the BiLSTM layer, and the overall feature of the sequences is obtained according to the degree of importance. Using the model, the experiment was carried out on the collected dataset. The results showed that the proved model can effectively solve the detection problem in DGA domain names and improve performance.
- The proposed model can handle both online and offline data input data. And the experiment results show that the model can classify wordlist-based domain names successfully that other state-of-the-art approaches could not identify.

The rest of the paper is organized as follows. In Sect. 2, we present generic background knowledge on DGA domain names detection, with an outline of related works. In Sect. 3 we describe the ensemble model developed to detect DGA domain names. The dataset and experimental evaluation of the proposed model are described thoroughly in Sect. 4. Finally, we conclude the paper and discuss possible future works in Sect. 5.

II. Background and Related Work

A. Domain Generation Algorithms

Over the last few years, most malware families have begun to use a different approach to communicate with their remote servers. Instead of using hardcoded server addresses, some malware families now use a Domain Generation Algorithm (DGA). DGA is a class of algorithm that takes a seed as an input, outputs a string and appends a top level domain (TLD) such as .com, .net [6]. DGA techniques vary in complexity, in order to combat the detection of malicious domain names based on features, some new DGAs simulate the composition and naming methods of normal domain names, which is called wordlist-based DGA domains, making the detection more difficult. Matsnu [7] pulls nouns and verbs from a built-in list of more than 1,300 words to form domains that are 24-character phrases. The malware suppobox [8] produces domains such as heavenshake.net, heavenshare.net, and leadershare.net, concatenating two pseudo-randomly chosen English dictionary words such as christinepatterson.net.

B. DGA detection methods

The research of distinguishing legitimate domain names from DGA domain names has been pursued for years. The legitimate domain names and the DGA ones are different in both structures and behaviors. By analyzing the behavior and structure of a domain name, one can determine whether it is a DGA domain name or not. Yadav [9] applied the statistical technique to detect algorithmically generated domain names, by modeling the time correlation and information entropy characteristics of successful domain names and failed domain names. Antonakakis [10] used a clustering approach on the length, level of randomness and character frequency distribution, including the n-gram distribution of observed domain names. Then a Hidden Markov model was used for determining the likelihood of a domain name to be a DGA one.

Further research work on the detection on DGA is developing towards the more and more extensive use of machine learning technology. Zang [11] proposed a detection algorithm by using cluster correlation that identifies the domain names generated by a DGA or its variants. Features such as TTL, the distribution of IP addresses, whois information and historical information from the domain names. Zhang [12] proposed a detection algorithm that analyzed the domain name features of character composition and the lexical hierarchical structure which included domain name length, character frequency and double letters to detect malicious domain names. Yadav [13] analyzed the KL distance, Jaccard coefficient and edit distance of DGA domain names. Bilge [14] proposed the EXPOSURE system by extracting 15 domain name features, used the J48 decision tree for classification. Raghuram [15] built a detection model on normal domain names for rapid identification of abnormal domain names, Grill [16] studied a method only through NetFlow information over DNS traffic rather than domain names, Wang [17] proposed using word segmentation to derive tokens from domain names to detect DGA domain names with features like the number of characters and digits. But in the actual network, features are difficult to extract and collect.

Meanwhile, deep neural networks have demonstrated their ability to find and extract relevant features as well as improved classification accuracy when compared to traditional machine learning methods. For the detection of DGA domain names, Woodbridge [3] showed that character-level long short-term memory networks (LSTMs) were extremely valid at detecting DGA domain names. In follow-up research, Tran [4] proposed a novel LSTM-based model in order to detect botnets that employ DGA to generate a large number of domains as a command and control server. Saxe [5] proposed a model based on CNN to detect DGA domain names. Anderson [18] investigated the use of adversarial learning techniques for the detection of DGAs. Shibahara [19] proposed a different algorithm that is using RNN on changes in network communication with a goal of reducing malware analysis time. These models based on deep neural networks had high accuracy in detecting DGA domain names with high randomness, but with a low identification rate to the DGA families that resemble English words, such as suppobox, which resulting in high false positives for normal domain names and reducing the credibility of a model.

Some recent progress has been made towards solving the issue of poor detection performance on wordlist-based DGA domains. Yang Liu [20] proposed a random forest classifier that used manually-extracted features such as word frequency, part-of-speech tags, and word correlations for classifying wordlist-based DGAs. Pereira [21] used a WordGraph method for detection of dictionary-based DGAs using graph-theory, their method outperforms convolutional neural networks for dictionary-based DGAs. Koh [22] proposed an approach that combines a pre-trained context-sensitive word embedding model ELMo with a simple fully-connected classifier to perform classification of domains based on word-level information. Curtin [23] proposed the measure smashword score which reflects how closely a DGA's generated domains resemble English words and build a machine learning model consisting of RNN using the generalized likelihood ratio test

(GLRT), also includes side information such as WHOIS information. The combined model was capable of effectively identifying DGA families with a high smashword score, such as the difficult matsnu and suppobox families.

C. Attention Mechanism

The attention mechanism in deep learning simulates the attention characteristics of the human brain, which can be understood as always paying Attention to more important information. Bahdanau [24] proposed applying the attention mechanism on neural network machine translation for the first time. The Attention mechanism has recently demonstrated success in a wide range of tasks such as speech recognition, machine translation, and Image recognition, It can be used alone or as a layer in other hybrid models. Assigning attention weights on neural networks has achieved great success in various machine learning tasks. Luong [25] designed two novel types of attention-based models for machine translation. Since then, more and more research has integrated attention mechanisms to text classification, relation Classification, and abstract extraction. Yang [26] proposed a hierarchical attention network for document classification, which has two levels of attention mechanisms applied both at the word and sentence-level. Ma introduced three kinds of temporal attention on different time steps. Rijke [27] using a two-level attention mechanism for summarization extraction. Zhou [28] proposed an attention-based bilingual representation model which learned the documents in both the source and the target languages and a hierarchical attention mechanism for the bilingual LSTM network. The model integrated with the attention mechanism is able to pay great attention to important content, words, and sentences in the surrounding context of a given input sequence. The successful application of attention mechanism in natural language brings sparks for detection of DGA domain names, mostly on wordlist-based ones.

III. Methodology

Based on the aforementioned discussion, we propose a model with attention mechanism for DGA detection and classification which is called ATT-CNN-BiLSTM model. In which the attention mechanism for domain names is introduced.

The architecture of the ATT-CNN-BiLSTM model with attention mechanism for detecting domain names is displayed in Fig 1. Which contains five components: embedding layer, CNN layer, BiLSTM layer, attention layer, and output layer. Before the output layer, we train domain name sequence and adopt drop-out strategy to avoid overfitting.

A. Embedding layer

Since the input sequence accepted by the neural network model is a fixed length vector and the domain name is a string, it is necessary to introduce the domain name vectoring step to convert the string into the input format of the neural network. Preprocessing is applied to the raw domain names. Namely converting the upper case characters to lower case primarily due to the fact that distinguishing the upper and lower case characters might end up in a regularization issue. Then the TLD is dropped, only the primary domain is left as the raw input of the model. The domain sequence can be denoted as $C_i = \{c_1, c_2, c_3, \dots c_n\}$, where n is the length of the domain. For example, the input domain name christinepatterson.net which is generated by suppobox would be expressed as *{c, h, r, i, s, t, i, n, e, p, a, t, t, e, r, s, o, n}* after preprocessing. The embedding layer only deals with strings of fixed length m. If the input string length is greater than m, the strings beyond m need to be truncated. When the input string length is less than m, the string would be padded. The embedding layer implements such a function, which projects sequences of input characters from the input domain to a sequence of vectors. Then the input string is encoded as a vector v of length d, which is a variable parameter.

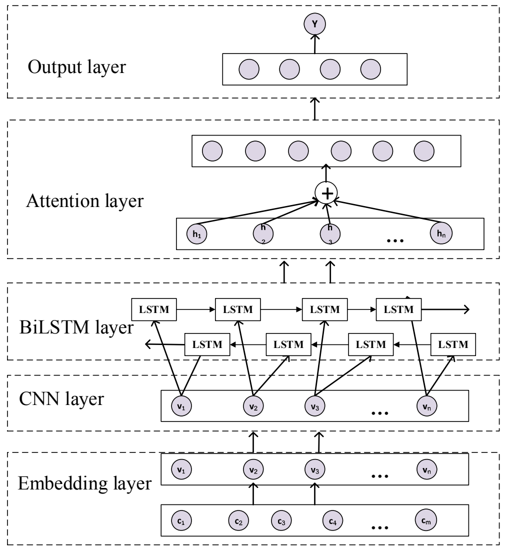

Fig. 1. The architecture of the ATT-CNN-BiLSTM model

B. CNN layer

After the embedding layer, the input sequence has been embedded into an $m \times d$ matrix, the next step is extracting locally detected features by applying the CNN layer which we use as our feature extraction component in our model. The t filters cover the entire length of the character embedding d, which is a sliding of convolution kernels over the sequence of character embedded. by using a filter of a width h and a non-linear function, the convolution operation can generate a new feature o_i as Equation (1).

$$o_i = f(\mathrm{W} \cdot v_i + \mathrm{b}) \tag{1}$$

Where W is the weight matrix, v_i is the embedding vector of a domain name, b is a bias term and f is a non-linear function. With the CNN layer, the similar sequential pattern of the domain names is detected. The CNN layer consists of the convolution layer and pooling layer. The convolutional layer has the characteristics of local connection and weight sharing, which can reduce the complexity of the model, while the pooling layer can reduce the size of parameters and prevent overfitting. Since the domain name is one-dimensional, we select convolution 1D which uses filter applying to the window of characters c to generate a new feature map and pooling 1D operation, which is a non-linear down-sampling is used to obtain the most significant feature as the next step.

C. BiLSTM layer

We use BiLSTM to get the contextual feature of each character in a long distance. In the bidirectional architecture, there are two layers of hidden nodes from two separate LSTMs, The two LSTMs capture the dependencies in different directions. The first hidden layers have recurrent connections from the last words while the second one's direction of recurrent of connections is flipped, passing activation backward in the sequences. Therefore, in the LSTM layer, we can get the forward hidden state from the forward LSTM network and the backward hidden state from the backward LSTM network. The two-state captures the compositional semantics information in both directions of the character sequences.

The BiLSTM consists of a forward and backward LSTM since the output vector of the CNN layer is $V = \{v_1, v_2, v_3, \dots v_z\}$, the forward LSTM read the input vector from v_1 to v_z , and the backward LSTM reads the input vector from v_z to v_1, namely the reverse order, meanwhile a pair of hidden states $(\overrightarrow{h_1}, \overrightarrow{h_2}, \dots, \overrightarrow{h_z})$ and $(\overleftarrow{h_1}, \overleftarrow{h_2}, \dots, \overleftarrow{h_z})$ are generated, we can get the output of the BiLSTM layer by combing the two hidden states as Equation (2).

$$h_i = \left[\overrightarrow{h_i}, \overleftarrow{h_i}\right]^T \tag{2}$$

D. Attention layer

As described above, some DGA generate domain names by concatenating words from a dictionary, such DGA families always follow a fixed pattern in the word combination. The distribution of single characters in domain names is no longer random and approximately normal, but the behavior of selecting words from the dictionary to generate domain name is random, which is reflected in the connection part of words. And many DGA detection models based on RNN represent the word sequences only using the hidden layer at the final node. In this paper, the hidden states at all the positions are considered with different attention weights. Since for DGA domain detection, focusing on some certain parts of the sequences will be effective to filter out the DGA irrelevant noise. We apply the attention mechanism to capture relevant features from the output of the BiLSTM layer, which is critical for prediction and needed to attend over while reading the input sequence and accumulating to a representation in the cell state. In order to capture the relationships between the current hidden state and all the previous hidden states, and to utilize the information from all the previous hidden states, we adopt general attention to capture the relationship between $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$, Each domain names consists of several words or characters which have the same weight. Many words carry noise or useless information, we trained weights for each character by attention mechanism to focus on the key features. The formulas to compute attention weight vector are:

$$\alpha_{ti} = v_a^T \tanh\left(\left[\overrightarrow{h_i}, \overleftarrow{h_i}\right]\right) \tag{3}$$

$$\alpha_t = \text{softmax}\left(\left[\alpha_{t1}, \alpha_{t2}, \dots \alpha_{t(t-1)}\right]\right) \tag{4}$$

$[h_1, h_2, \dots h_t]$ is the input matrix which is produced by BiLSTM layer. Then the context vector can be calculated based on the attention weight vector and the hidden states as Equation (5).

$$c_t = \sum_{i}^{t-1} \alpha_{ti} h_i \tag{5}$$

The attentional hidden state h' is calculated by Equation (6) , which is based on the current hidden state h_t and context vector c_t, where W_c is the weight matrix of attention layer, A weight vector could learn word features automatically and record the significant information in a domain. A domain feature can be represented by multiplying the weight vector. and r is the dimensionality of attention state. Also, dropout is used in the dense layer to avoid overfitting.

$$h' = \tanh(W_c[c_t; h_t]) \tag{6}$$

E. Output Layer

In the output layer there are two dense layers. The output of the first dense layer is fed into the second dense layer with n hidden neurons, where n is the class number of domain names. Then, we feed the attention vector after dropout into the softmax function for prediction. w_s and b_s are the parameters to be learned. The softmax function is chosen as the activation function, which calculation result is between 0 and 1.

$$\text{p} = \text{softmax}\left(\left[w_s h'' + b_s\right]\right) \tag{7}$$

From the above model, we get the probability p, which will learn directly from the features to determine whether a domain name is benign or DGA on the detection task, and which domain family a domain name belong to on the classification task.

IV. Experiments

This section starts with describing the necessary details of the benign domain names and DGA domain names dataset, the comparison methods, and the experiment design in the paper.

A. Description of dataset

The labeled domain name dataset used in this paper include benign domain names and DGA ones. The benign domain names were extracted from Alexa [29]. We downloaded Top 500,000 domain names in Alexa. Alexa ranks websites based on their popularity in terms of the number of page views and number of unique visitors. The DGA domain names were collected from two complementary sources. Namely John Bambenek [30] and 360netlab [31]. The total amount of DGA domain names is 300,046, and they correspond to 19 different malware families. Including Cryptolocker, locky and suppobox, etc. The DGA families can fall into three categories: the arithmetical-based, the wordlist-based, and the part-wordlist-based schema. The former schema usually calculates a sequence that has a direct ASCII representation usable for a domain name. The wordlist-based one consists of concatenating a sequence of words from one or more word lists. While the part-wordlist-based schema means the algorithm combines the wordlist-based and the arithmetical-based one, such as banjori and beebone. The domain name generated is like “bnwgbyplay.com”, where the first six characters “bnwgby” are arithmetical-based and the last four characters “play” are wordlist-based. In the dataset, 80% of data is used for training, and the rest 20% is used for testing. The data distribution is shown in Table I.

TABLE I. SUMMARY OF THE COLLECTED DATASET

Domain Type	Schema	Sample
Alexa	/	500000
banjori	part-wordlist-based	25000
beebone	part-wordlist-based	210
cryptolocker	arithmetical-based	6000
dyre	arithmetical-based	823
emotet	arithmetical-based	20000
gameover	arithmetical-based	16615
locky	arithmetical-based	8028
matsnu	wordlist-based	20694
murofet	arithmetical-based	26520
necurs	arithmetical-based	21843
Post	arithmetical-based	22000

pykspa_v1	arithmetical-based	23293
qakbot	arithmetical-based	26058
ramnit	arithmetical-based	24500
rovnix	arithmetical-based	25800
suppobox	wordlist-based	10055
tinba	arithmetical-based	19766
urlzone	arithmetical-based	1845
volatile	part-wordlist-based	996

B. *Experimental setup*

In our experiments, we considered TensorFlow in conjunction with Keras as software framework. To increase the speed of gradient descent computations of deep learning architectures, we use with GPU enabled TensorFlow in one Nvidia Tesla P100. By constantly adjusting and optimizing the parameters in our experiments, the most effective hyperparameters are set as follows:

- The dimension of the embedding vector was set to 128 in the embedding layer.
- The neural models were trained using a batch size of 128 on the training set.
- The number of hidden layer nodes in BiLSTM model was set to 128.
- The filters of the CNN layer was set to 64.
- RMSProp was employed as an optimization algorithm.
- The dropout rate was set to 0.2.

C. *Comparison with Baseline Methods*

We set three baseline methods to compare with the proposed ATT-CNN-LSTM model. The models in the experiment was as follows.

1) *LR model*

features are first extracted by using term frequency-inverse document frequency (TF-IDF), then we use Logistic Regression (LR) which is a classical model in machine learning to classify the domain names.

2) *LSTM model*

LSTM model proposed by [5] was adopted as a comparison model in our paper, which used only the LSTM layer to extract features and classify the domain names.

3) *CNN-BiLSTM*

CNN-BiLSTM is a model that attention mechanism is not included, while the remaining layer and parameter setting are same as AB-BiLSTM model.

4) *ATT-CNN-BiLSTM model*

ATT-CNN-BiLSTM is the model we proposed to detect DGA domain name in this paper.

D. *Experiment Design*

In order to evaluate the performance of the models, we used a 10-fold cross-validation strategy over the dataset. First, the dataset is split into 10 folds, then the nine folds are trained and the remaining one is used for verification. This process uses each fold for validation once and is repeated ten times. Finally, all the metrics on validation folds are averaged and a better estimate of the performance is achieved.

We use standard accuracy, precision, recall, and F1-score as the classification evaluation metrics to evaluate the performance of malicious domain names detection. The formula of the metrics can be defined as Equation (8)-(11).

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{8}$$

$$precision = \frac{TP}{TP+FP} \tag{9}$$

$$recall = \frac{TP}{TP+FN} \tag{10}$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{11}$$

where TP is true positives, TN is True Negatives, FN is False Negatives and FP is False Positives respectively. We define the malicious domain names as positive and the benign ones as negative. Since the classes in the dataset are imbalanced, we also used the Receiver Operating Characteristic (ROC) curve to evaluate Area Under the Curve (AUC) statistic.

V. Experimental Results

In this section, we present the evaluation results of the experiment and analyze the result of each model.

A. *The task of Detection and Classification*

- Detection task

Fig 2 displays the ROC curves of each model. The AUC value of ATT-CNN-BiLSTM is 0.9990, the results show that the model we proposed in the paper has performed better in distinguishing the domain names as either benign or malicious in comparison to LR, LSTM and CNN-BiLSTM model.

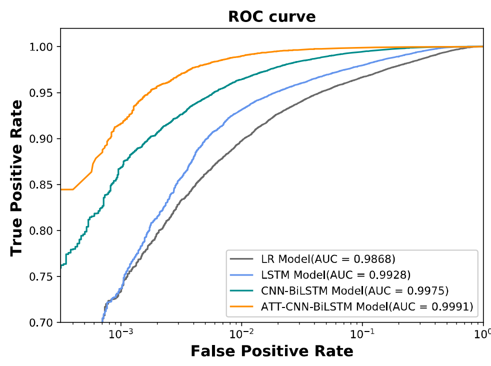

Fig. 2. ROC curve on the detection of DGA domain names

The comparisons of accuracy, precision, recall and F1 score of the methods are displayed in Fig 3, the model based on the attention mechanism outperforms the other three models on the accuracy, precision, recall, and F1 score. The ATT-CNN-BiLSTM model has better detection result on F1-score of 0.9872 than the traditional LR model that using statistical characteristics, which indicates that the new DGA domain name evaded the traditional statistical characteristics, and the LR model is not suitable for the detection of such domain names. The recall and precision of the ATT-CNN-BiLSTM model are higher than those of the best performing LSTM and CNN-BiLSTM model in the traditional method, respectively, indicating that the attention mechanism can better detect DGA domain names and improve the overall detection effect.

For the accuracy and precision measures, on average, the ATT-CNN-BiLSTM model was 3% higher than the feature-engineer model with LR, the recall is nearly 5% higher than the LR model. On the one hand, the feature-based model relies on other natural language processing tools, the accumulated errors have a great impact on performance and the effective feature extraction is insufficient. On the other hand, external semantics such as word frequency have limited effects on the DGA domain name detection task especially the wordlist-based DGA domain name, while neural networks can encode semantic information into high-dimensional hidden feature space and extract more features.

Compared with the LSTM model, the F1 value of ATT-CNN-BiLSTM model is 2% higher. The CNN-BiLSTM model also has higher accuracy and F1 score than LSTM model. This is because both the CNN-BiLSTM and ATT-CNN-BiLSTM model combine the advantages of CNN and BiLSTM, which can capture the local features and long-distance dependence information within the domain name sequences.

Compared with the CNN-BiLSTM model, the F1 value of ATT-CNN-BiLSTM model is 1 percent higher. Since the two models differ only in the attention layer, which can give relatively large weight to important features such as the joint part of several words of domain names. This indicates that the attention mechanism is effective for the DGA domain name detection task and can improve the overall detection effect.

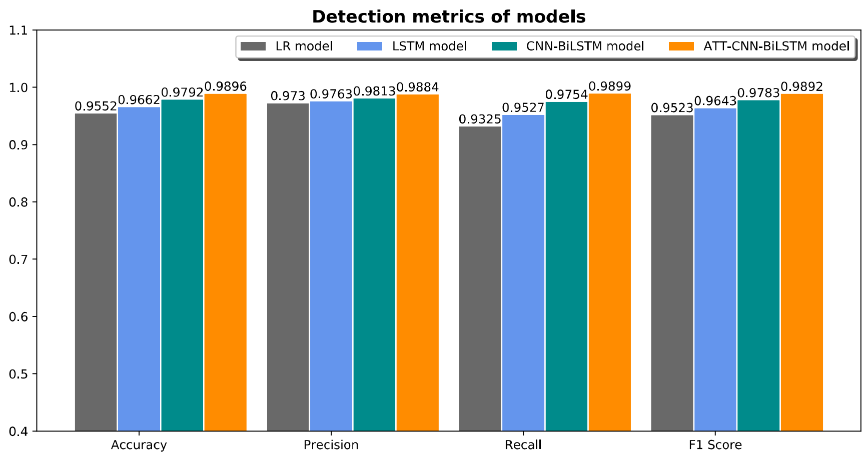

Fig. 3. Evaluation results on the detection of DGA domain names

- Classification task

The same multiclass classification experiment was run on the dataset. For the classification task, the metrics we used in this paper were the precision, recall and F1 scores. Since the precision, recall and F1 scores are class-specific measures, they must be averaged to yield a global result. Micro-averages sum up the individual TP, FP, and FN for all classes, while macro-averages take the mean of the individual scores for all classes. In other words, macro-averages treats all DGA families equally, while micro-averaging favors the class with more samples. As shown in Table II, the ATT-CNN-BiLSTM model showed the best metrics on both micro-averages and macro-averages (where the micro-average of F1-score was 0.89 and macro-average of F1-score was 0.81).

B. Analysis and discussions

We can refine the analysis of the experiment results by looking at detection and classification results for each malware family. Compared to the LR and LSTM model, the F1 score of CNN-BiLSTM model reached 0.9887 and the macro average was 0.72, which proved that with CNN layer we could extract locally features effectively. And the proved ATT-CNN-BiLSTM model got the best metrics of accuracy, precision, recall and F1 score in all the models, which proved the good capability of extracting common patterns in domain names. In the schema of arithmetic-based DGA, take dyre and urlzone malware, for example, the F1 score got more than 92% with a small size less than 2000. In the part-wordlist-based DGA, since the random strategy was also applied in them, the ATT-CNN-BiLSTM model showed better detection efficiency on such schema such as banjori and volatile malware, which reached 0.99 of the F1 score.

As to wordlist-based DGA families, the distribution of single characters in domain names is no longer random and approximately normal. The behavior of selecting words from the dictionary to generate a domain name is random, which is reflected in the connection part of words. The proposed ATT-CNN-BiLSTM model can capture such random by the integrated attention mechanism. In case of suppobox, the F1 score was 0.91 compared to 0.46 for the bigram model and 0.33 for CNN-BiLSTM model, and to matsnu malware, we got a F1 score of 0.81, which was better than the other three models. After integrating the attention mechanism into the neural model, the ability to learn the combination pattern of the word lists employed by the DGA was promoted to a certain level.

TABLE II. PRECISION, RECALL AND F1 SCORE FOR CLASSIFICATION

Domain Type	**LR**			**LSTM**			**CNN-BiLSTM**			**ATT-CNN-BiLSTM**			**Support**
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
benign	0.94	0.99	0.96	0.95	0.98	0.97	0.94	0.99	0.96	0.99	0.98	0.99	99999

banjori	0.97	0.98	0.98	0.99	1.00	0.99	1.00	0.99	0.99	1.00	1.00	1.00	5012
beebone	1.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.98	1.00	0.99	42
Cryptolocker	0.00	0.00	0.00	0.26	0.24	0.25	0.21	0.02	0.03	0.28	0.27	0.28	1200
dyre	0.95	0.99	0.97	0.99	1.00	0.99	0.99	0.98	0.99	0.99	1.00	1.00	167
emotet	0.65	0.81	0.72	0.66	1.00	0.79	0.66	1.00	0.79	0.65	0.81	0.72	3985
gameover	0.34	0.11	0.16	0.90	0.00	0.01	0.53	0.01	0.02	0.35	0.29	0.31	3323
locky	0.26	0.03	0.06	0.67	0.00	0.00	0.14	0.00	0.00	0.43	0.26	0.32	1610
matsnu	0.27	0.26	0.27	0.69	0.86	0.77	0.76	0.78	0.76	0.75	0.87	0.81	4133
murofet	0.96	0.99	0.98	0.95	1.00	0.97	0.97	0.99	0.98	0.98	1.00	0.99	5304
Post	0.59	0.73	0.66	0.78	0.73	0.76	0.68	0.87	0.76	0.83	0.76	0.79	4373
necurs	0.24	0.22	0.23	0.52	0.32	0.40	0.56	0.19	0.29	0.61	0.66	0.63	4400
pykspa_v1	0.9	0.88	0.89	0.92	0.96	0.94	0.98	0.93	0.96	0.98	0.98	0.98	4662
qakbot	0.64	0.41	0.50	0.71	0.64	0.67	0.62	0.72	0.67	0.75	0.65	0.70	5217
ramnit	0.38	0.49	0.43	0.60	0.71	0.65	0.62	0.71	0.66	0.60	0.75	0.67	4900
rovnix	0.85	0.90	0.87	1.00	0.99	1.00	1.00	0.99	0.99	1.00	0.99	1.00	5160
suppobox	0.75	0.33	0.46	0.87	0.20	0.33	0.99	0.09	0.17	0.87	0.94	0.91	2011
tinba	0.59	0.80	0.68	0.67	0.96	0.79	0.77	0.98	0.86	0.91	0.98	0.94	3955
urlzone	0.86	0.75	0.80	0.96	0.91	0.93	0.98	0.91	0.94	0.98	0.92	0.95	369
volatile	0.98	1.00	0.99	0.97	0.69	0.81	1.00	0.36	0.53	0.99	0.99	0.99	185
micro avg	0.80	0.80	0.80	0.86	0.86	0.86	0.87	0.87	0.87	0.89	0.89	**0.89**	120007
macro avg	0.66	0.63	0.63	0.69	0.65	0.66	0.72	0.73	0.72	0.81	0.81	**0.81**	120007

But there are still a few exceptions, Table II also shows that malware Cryptolocker, gameover and locky are not properly classified. In the detection task, the ATT-CNN-BiLSTM model can distinguish the domain names generated by Cryptolocker gameover and locky from the normal ones, but in the classification task, all models have poor performance over these malware classes though ATT-CNN-BiLSTM gets the highest classification accuracy. The reason is these malware use a series of multiplies, divisions, and modulus based on a single seed to generate DGA domain names. The unigram distribution of Cryptolocker and locky is shown in Fig 4, which looks exactly the same, while the same distribution of characters in domain names causes the misclassification on the two similar classes. A large proportion of the incorrectly classified Cryptolocker DGA were classified as locky ones. Meanwhile, the gameover and Post malware have a similar situation as showed in Fig 5, which leads to poor classification effect on the gameover malware.

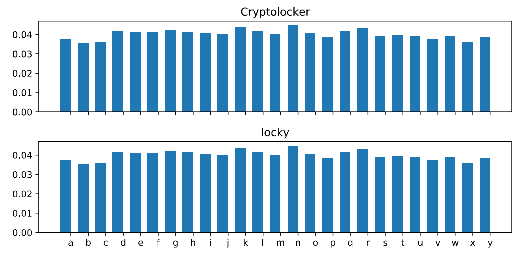

Fig. 4. Unigram distributions for Cryptolocker and locky

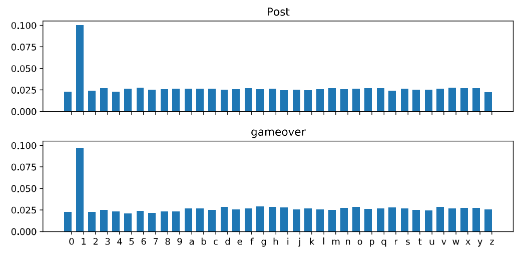

Fig. 5. Unigram distributions for gameover and Post

VI. Conclusions

In this paper we have considered the problem of DGA domain detection, since many malware imitate the pattern of normal domain names by concatenating pseudo-randomly chosen English dictionary words, to generate domain names and achieve the effect of concealment and confrontation. We proposed an effective ATT-CNN-BiLSTM model which integrated attention mechanism and deep neural network to detection and classification the domain names. Compared with the statistical characteristics and most widely used LSTM model, the ATT-CNN-BiLSTM model could improve the accuracy of detection task without handcrafted feature extraction and achieved better performance on arithmetic-based, part-word-based and wordlist-based DGA such as matsnu and suppobox families. in the classification task. Experiment results show the effectiveness of the model, by which we get F1 score of 98.92% on average for the detection and F1 score of 81% on macro average for the classification of domain names. And the results prove that attention mechanism can improve the identification validity of the wordlist-based DGA domain names.

Future work will be focused on the integration of this neural model as part of a larger machine-learning architecture for the detection of cyber-threats in real traffic data. We also intend to add side information to improve classification accuracy and differentiate the DGA classes that generate similar domain names.

Acknowledgment

This work is supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDC02030200) and the National Natural Science Foundation of China (Grant No. 61572481).

References

[1] Andrews, Mark. "Negative caching of DNS queries (DNS NCACHE)." 1998.

[2] Y. Zhauniarovich, I. Khalil, T. Yu, and M. Dacier, "A survey on malicious domains detection through DNS data analysis," ACM Computing Surveys, vol. 51, no. 4, 67:1–67:36, 2018.

[3] J.Woodbridge, H.S. Anderson, A.Ahuja, and D.Grant, "Predicting domain generation algorithms with long short-term memory networks," arXiv:1611.00791 [cs.CR], 2016.

[4] Hieu Mac, Duc Tran, Van Tong, Linh Giang Nguyen, and Hai Anh Tran. "DGA Botnet Detection Using Supervised Learning Methods."

In Proceedings of the Eighth International Symposium on Information and Communication Technology, SoICT 2017, pages 211–218.

[5] J. Saxe, K. Berlin, “eXpose: A character-level convolutional neural network with embeddings for detecting malicious URLs file paths and registry keys”, 2017.

[6] D. Plohmann, K. Yakdan, M. Klatt, J. Bader, E. Gerhards-Padilla, "A Comprehensive Measurement Study of Domain Generating Malware", Proceedings of the 25th USENIX Security Symposium (SECURITY), 2016.

[7] Stanislav Skuratovich. “Matsnu technical report.” http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/, 2015.

[8] J. Geffner, “End-to-end analysis of a domain generating algorithm malware family.” Black Hat USA 2013.

[9] S. Yadav, A. K. K. Reddy, A. N. Reddy, S. Ranjan, "Detecting algorithmically generated domain-flux attacks with DNS traffic analysis", Networking IEEE/ACM Transactions on, vol. 20, no. 5, pp. 1663-1677, 2012.

[10] *M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, “Building a dynamic reputation system for DNS.,” in USENIX security symposium, 2010. pp. 273-290.*

[11] X. Zang, J. Gong, and X. Hu, “Detecting malicious domain name based on AGD,” Journal on Communications, vol. 39, no. 7, pp. 15–25, 2018.

[12] Y. Zhang, Y. Zhang, and J. Xiao, “Detecting the DGA-based malicious domain names,” in Proceedings of the International Standard Conference on Trustworthy Computing and Services, Beijing, China, November 2013. pp. 130-137.

[13] S. Yadav, A. K. Reddy, A. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in Conference on Internet Measurement (IMC), 2010.

[14] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, C. Kruegel, "Exposure: A passive DNS analysis service to detect and report malicious domains", ACM Trans. Inf. Syst. Secur., vol. 16, no. 4, Apr. 2014.

[15] J. Raghuram, D. J. Miller, G. Kesidis, "Unsupervised low latency anomaly detection of algorithmically generated domain names by generative probabilistic modeling", J. Adv. Res., vol. 5, no. 4, pp. 423-433, 2014.

[16] M. Grill, I. Nikolaev, V. Valeros, and M. Rehak, “Detecting DGA malware using NetFlow,” in Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manag. (IM), Ottawa, ON, Canada, 2015, pp. 1304–1309.

[17] Wang W, Shirley K: “Breaking bad: detecting malicious domains using word segmentation.” 2015.

[18] H. S. Anderson, J. Woodbridge, B. Filar, "DeepDGA: Adversarially-tuned domain generation and detection" in Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security., ACM, 2016.

[19] T. Shibahara et al., "Efficient dynamic Malware analysis based on network behavior using deep learning", Proc. IEEE Global Commun. Conf. (GLOBECOM), pp. 7-18, Feb. 2017.

[20] L. Yang, G. Liu, J. Zhai, Y. Dai, Z. Yan, Y. Zou, and W. Huang, “A novel detection method for word based DGA,” in Proceedings of the 4th International Conference on Cloud Computing and Security, vol. 2, 2018. pp. 472-483.

[21] M. Pereira, S. Coleman, B. Yu, M. De Cock, and A. Nascimento, “Dictionary extraction and detection of algorithmically generated domain names in passive DNS traffic,” in Proceedings of the 21st International Symposium on Research in Attacks, Intrusions, and Defenses, 2018, pp. 295-314.

[22] Koh J J, Rhodes B. “Inline Detection of Domain Generation Algorithms with Context-Sensitive Word Embeddings.” in Proceedings of IEEE International Conference on Big Data. ”2018. 2966-2971.

[23] R. R. Curtin, A. B. Gardner, S. Grzonkowski, A. Kleymenov, A. Mosquera, "Detecting DGA domains with recurrent neural networks and side information", 2018.

[24] D. Bahdanau, K. Cho, Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", International Conference on Learning Representations, 2015.

[25] T. Luong, H. Pham, C. D. Manning, "Effective approaches to attention-based neural machine translation", Proc. Empirical Methods Natural Lang. Process., pp. 1412-1421, 2015.

[26] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical attention networks for document classification,” in NAACL,2017, pp. 1480–1489.

[27] M. D. Rijke, M. D. Rijke, M. D. Rijke, M. D. Rijke, M. D. Rijke, M. D. Rijke, "Leveraging Contextual Sentence Relations for Extractive Summarization Using a Neural Attention Model", International ACM SIGIR Conference on Research and Development in Information Retrieva1, pp. 95-104, 2017.

[28] Zhou X, Wan X, Xiao J. “Attention-based LSTM network for cross-lingual sentiment classification.” In Proceedings of the 2016 conference on empirical methods in natural language processing. Austin: Association for Computational Linguistics;2016, pp. 247-56.

[29] Alexa. The web information company. http://www.alexa.com/, 2007.

[30] Bambenek Consulting. http://osint.bambenekconsulting.com/feeds, 2019.

[31] 360netlab, https://data.netlab.360.com/dga/, 2019.