ECAI 2020
G.D. Giacomo et al. (Eds.)

doi:10.3233/FAIA200301

OpenSMax: Unknown Domain Generation Algorithm Detection

Yao Lai[1] and **Guolou Ping**[1] and **Yuexin Wu**[1] and **Chenhui Lu**[1] and **Xiaojun Ye**[1,2]

Abstract. Botnet has become one of the most frequent attack patterns in cyberspace, and most of them are concerned with Domain Generation Algorithms (DGAs). Therefore, many researchers have proposed various machine learning models for DGA domain name detection, but how to detect unknown classes of DGA domain names (unknown DGAs) is still a challenging problem. In fact, the problem of detecting unknown classes is also called open set recognition problem. To tackle this issue, we propose a novel classification model OpenSMax which can not only detect various DGA domain names but also classify them into known and unknown classes of DGAs. In this model, we use the one-hot encoding method and the Long Short-Term Memory (LSTM) model to extract the features of the Top Level Domain (TLD) and the Second Level Domain (SLD) respectively. Then, these two feature categories are concatenated and propagated forwards by two fully connected layers for known DGA domain name detection and classification. Finally, both the openmax layer (the layer before the softmax layer) and the softmax layer are used to build One-Class Support Vector Machine (SVM) models for unknown classes recognition. In our experiments, OpenSMax model outperforms the state-of-art methods both in known and unknown DGA domain names detection tasks. Also, OpenSMax provides a bounded open space risk in theory, and therefore it formally provides an effective solution for unknown DGA domain name detection.

1 INTRODUCTION

Botnet is one of the main ways of cyberattacks and it has posed serious threats to cyberspace security. A typical botnet architecture usually includes bots, Command and Control (C&C) servers and a botmaster [1]. Firstly, a botmaster will create a malware instance and send it to bots. Then, when bots have been under the control of the malware, the botmaster will send commands to C&C servers, and then C&C servers will forward them to related bots [2]. Finally, bots will execute commands to implement various attacks such as denial-of-service attacks, sending spam mails and so on.

In order to avoid being detected by the network defense system such as intrusion detection systems (IDS), the botmaster usually need to use variable IP addresses and domain names to obfuscate network traffic when communicating with controlled bots. Specifically, these domain names are usually generated by Domain Generation Algorithms (DGAs) which can dynamically generate a series of DNS domain names to bypass the blacklist of IDSs, and the botmaster can use these DGA domain names to establish connections with malicious bots. This technology is also called as Domain Fluxing [3].

To deal with Domain Fluxing, many security experts have proposed DGA domain name detection methods. Some methods are derived from network traffic analysis [4, 5]. However, these methods are based on the assumption that most of botnet attacks have similar network traffic patterns and people need the access authorization of DNS servers which is not easy to get for monitoring DNS traffic if they want to implement these solutions. As a result, many researchers attempt to detect botnet attacks by detecting DGA domain names [1, 6]. However, most of proposed learning models can only detect known classes of DGA domain names because they are trained on known DGAs training set. Unfortunately, the number of DGAs is infinite and none of the existing datasets can cover real DGA generated domain names. Therefore, the most critical issue is how to deal with DNS domain names generated by known and unknown DGAs at the same time.

The problem of detecting unknown classes is called Open Set Recognition problem [7], which is an significant issue in data mining field [8, 9]. It describes a scenario where some classes are unseen in the training set but appear in the testing set, and the learning model needs not only to accurately classify the known classes but also to effectively deal with the unknown ones [10]. In this paper, we define unknown DGA classes as classes which do not appear in the training phase. Abhijit et al. [11] proposed OpenMax model to solve unknown class detection issue. It uses a deep learning model to classify known classes and then uses an outlier detection model to detect unknown classes. However, this method is proposed for the image classification task and its effectiveness is not verified in experiments for unknown classes of DGA domain name detection task.

In this paper, we propose a DGA domain name detection model OpenSMax, which can effectively detect both known and unknown DGA domain names. First, we train a classification model to detect and classify known DGA domain names. Then, inspired by the OpenMax model [11], we extract the activation value from both the openmax layer (the fully connected layer before the softmax layer) and the softmax layer from the classification model with correctly classified training data. As a result, each domain name is mapped into a point in the 2D plane space, where the two dimensions represent the activation values from openmax and softmax layers. Finally, we use the extracted activation values to fit a One-Class SVM model which is used to generate a hyper-plane where most of correctly classified points are located. In the testing phase, we first get the output label from the classification network. And then we use the corresponding One-Class SVM model to find whether the sample is out of the hyper-plane. If it is out of the hyper-plane, we recognize it as an outlier and classify this domain name into unknown classes of DGA

[1] School of Software, Tsinghua University, China
emails: laiy17@mails.tsinghua.edu.cn, pgl19@mails.tsinghua.edu.cn, wuyuexin333@gmail.com, luch18@mails.tsinghua.edu.cn, yexj@mail.tsinghua.edu.cn
[2] National Engineering Laboratory for Big Data System Software, Tsinghua University, China

domain names. Otherwise, we maintain the initial output label from the classification model.

In summary, we make the following contributions:

- We use different methods to extract TLD and SLD features for the reason that different parts of domain name have different characteristics. Compared with the methods using the same way for TLD and SLD feature extractions, our method achieves 1.22% higher accuracy in known DGA domain name detection.
- We use both the openmax and softmax layer of the classification model for unknown class detection, which can take the relationship between the activation values of different layers into account. Compared with the method only using the openmax layer, our method by using two layers achieves 1.84% higher accuracy when half of DGA classes are unknown classes.
- We propose the OpenSMax model and prove that our model has a bounded open space risk [12]. Therefore, it formally provides an open set recognition solution in theory. Experiments show that OpenSMax achieves 83.62% accuracy when half of DGA classes are unknown classes, which is 14.28% higher than the same classification method without unknown detection phase and 6.00% higher than the start-of-art open set recognition method OpenMax, 9.50% higher than open set recognition method DOC.

2 RELATED WORK

2.1 Known DGA Domain Name Detection

In order to detect DGA domain names, some methods are proposed based on the low-level DNS traffic. They use not only domain names but also other information such as WHOIS data, traffic frequency and the packet contents for DGA domain name detection. The DBod method [4] has been proposed to analyze the query behaviors of the DNS traffic for DGA domain name detection, and the query behaviors are clustered from the correlation topology of the network traffic. Pleiades method [5] analyzes streams of unsuccessful domain name resolutions to identify DGA-based botnets. These methods utilize low-level traffic information, which is not easy to get from the DNS servers in most cases.

To achieve real-time detection, many domain name based DGA detection methods have been proposed. These methods rely on the assumption that most of the normal domain names are combined of meaningful, readable words, while DGA domain names mostly look like randomly generated strings. Therefore, many methods [13, 14, 15] extract length, n-gram frequency and meaningful word ratio as features and use machine learning methods to detect DGA domain names. Furthermore, many deep learning models have been proposed without the manual feature engineering. For example, the model [6] based on LSTM method receives the better accuracy than the n-gram features and manual features based machine learning models. The CNN model called n-CBDC [1] can reach higher detection accuracy than other deep learning models on some DGA domain names. However, all these methods can only detect known DGA domain names. If we extend these methods to unknown DGA domain name detection, the detection accuracy will be degraded significantly.

2.2 Open Set Recognition

Compared with closed set, Open Set Recognition was usually overlooked in the past [10]. To solve this problem, in early times, a lot of SVM-based models were proposed by modifying SVM model's loss function and calculate process, such as 1-vs-Set machine [12] and Weibull-calibrated SVM (W-SVM) [16]. They are all derived from the One-Class SVM model [17] and the aim of this model is to generate a hyper-plane to make sure that most of samples are inside this hyper-plane. During the testing phase, if one sample is out of the hyper-plane, it will output an outlier label. However, all these SVM-based models need feature engineering on the raw data.

For this reason, some deep network based models were proposed. OpenMax [11] is the first open set classification model on computer vision datasets. It extracts the value of the openmax layer of deep network to fit a Weibull distribution for each class. In the testing phase, it can readjust the value of the output vector of softmax layer according to the parameters of the distribution and give a redistributed probability for each known or unknown class. DOC [18] is an open set classification model on text datasets. It uses the sigmoid activation function to replace softmax function and uses the output of the sigmoid function to fit Gauss distribution for unknown class detection. Based on Generative Adversarial Network (GAN), G-OpenMax model [19] was proposed. It can generate samples of unknown classes, but it has no significant improvement on real image classification. In addition, by reconstructing the feature of the samples, Classification-Reconstruction learning for Open-Set Recognition (CRSOR) [20] was proposed for image recognition. It uses deep hierarchical reconstruction net to extract hidden features from the image data, and recognizes unknown classes with these features. In general, these methods are applied in the computer vision and text areas. As a result, the effectiveness of these methods on DGA domain name detection has not been verified in experiments. Based on existing methods, we give a more suitable solution for DGA domain name detection.

3 PROBLEM STATEMENT

A domain name can be seen as a *string* which is divided by *dot* (.) into two or more parts. The right-most part is the Top-Level Domain (TLD), and the part on the left of the TLD is the Second-Level Domain (SLD) and so on. Formally, an English domain name is case-insensitive and can only be taken from the following character set C:

$$C = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, \\ k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, -, .\}$$

A domain name with length n can be represented as $d(n) = (d_1 d_2 ... d_i ... d_n), d_i \in C$. The length of domain n cannot exceed 255. There must exist an index k, $s.t.\ d_k =' .'$. If there exist more than one index, we select the maximum index as k. In this paper, we define $(d_{k+1} d_{k+2} ... d_n)$ as the TLD part and $(d_1 d_2 ... d_{k-1})$ as the SLD part. E.g., the domain name *example.com* consists of the TLD *com* and the SLD *example*.

Domain names can be divided into many classes according to different classification methods. In this paper, we classify domain names in a hierarchy structure as Figure 1. First, we divide the domain names into benign domain names and DGA domain names. Then, we divide DGA domain names into two classes based on known and unknown DGAs. For known classes of DGA domain names, we can also split them into different sub-classes (different DGAs). There is a class hierarchy in domain name classes, which can be seen in Figure 1. For our domain name detection problem, the input is a domain name string including both TLD and SLD and the output is the class label to which the domain name belongs. Assuming there are N known DGAs, we use label 0, $\{1, 2, 3, ..., N\}$ and

$N+1$ to represent the benign domain name class, known classes of DGA domain names and unknown classes of DGA domain names. Therefore, our problem is to fit a function to correctly output the label of the inputted domain name.

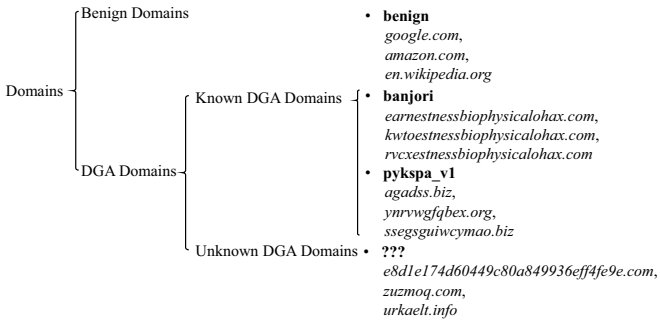


Figure 1. DNS Domain Name Classification

4 OPENSMAX MODEL

4.1 Overview

The overall architecture of our proposed OpenSMax model is shown in Figure 2 and it can be divided into two phases. The first phase is Known Class Detection Phase and it is used to recognize known classes of DGA domain names and benign domain names. Meanwhile, the activation values of the classification model's softmax layer (last layer) and openmax layer (penultimate layer, the layer before the softmax layer [11]) are extracted for building a 2D feature space, where two dimensions represent the maximum activation values from openmax and softmax layers. The data in this feature space are used as the input of the second phase. The second phase is Unknown Class Detection Phase. It consists of One-Class SVM models and each model is built for each class based on outputs extracted from the first phase. If one input is recognized as an outlier by the corresponding One-Class SVM model, this input should be recognized as the DGA domain name from an unknown DGA class. In addition, the mean activation value can be used to exclude some wrong unknown class detection results. Only if all dimensions of the output vector in the first phase are less than the corresponding mean values, it will be inputted into the unknown class detection phase.

4.2 Known Class Detection Phase

In this paper, the known class detection phase is used to classify the benign domain names and DGA domain names without recognizing unknown DGA domain names. Jonathan Woodbridge et al. [6] points out that people can use Long Short-Term Memory networks to detect DGA domain names, the performance of which is much better than Hidden Markov Model, bigram model and manual features model. However, most of these models do not take the TLD into account [6] or treat the TLD and SLD equally [21]. In fact, the top domain contains an amount of information for DGA detection. Former study [2] points out that most kinds of DGAs have their own sets of TLDs to select. Also, the number of TLDs is very limited. According to related survey, there are about 1,000 TLDs in Internet addresses. Because most of the TLDs are not commonly used, there are in total 336 TLDs in our dataset. In this situation, it is more suitable to use One-hot encoding to represent the features rather than using LSTM model. So, we propose the model which combines the One-hot encoding model and LSTM encoding model to extract the TLD and SLD features.

As Figure 2 shows, we divide one domain name into two parts: SLD and TLD. (According to our problem statement, if there exist more than two levels of domain names, we regard the third and higher levels of domain names as a part of SLD.) For SLD, we expand it by using zero padding to the fixed length and input it into the LSTM model to extract encoding features. For TLD, we encode it to one-hot feature vector and input it into two Fully Connected (FC) layers. After that, we concatenate the outputs of the TLD and SLD features and input it into another two FC layers. Taking Figure 2 as an example, the input domain name is *google.com* and it is a benign domain name. According to the definition, the SLD is *google* and the TLD is *com*. We use the padding character to expend the SLD *google* to the fixed length (31 in our experiments) and input it into the LSTM model to get the encoded features. Meanwhile, we encode the TLD *com* into the one-hot feature vector and input it into two FC layers. Then we concatenate the outputs of these two parts and input it into another two FC layers. When our model has been trained, we can find that the output in the known class detection phase is Class 0 (benign) as Figure 2.

4.3 Unknown Class Detection Phase

As shown in Figure 2, the unknown class detection phase uses the outputs of both softmax layer and openmax layer in previous phase as the input. The calculation process is shown in Algorithm 1. We assume that there are in total N+1 known classes (including 1 benign class and N classes of DGA domain names) and the k-th training sample is $\boldsymbol{x}^{(k)}$. Then, we define that the output of softmax layer (it can also be regarded as the value of the vector-valued function) as $\boldsymbol{v}(\boldsymbol{x}^{(k)}) \in \mathbb{R}^{(N+1)\times 1}$ and the output of openmax layer as $\boldsymbol{u}(\boldsymbol{x}^{(k)}) \in \mathbb{R}^{(N+1)\times 1}$. Obviously, $u_j(\boldsymbol{x}^{(k)})$ means the j-th elements of the vector $\boldsymbol{u}(\boldsymbol{x}^{(k)})$. For each training data $(\boldsymbol{x}^{(k)}, y^{(k)})$, if the classification model of the known class detection phase correctly predicts it as class j, the corresponding output $(u_j^{(k)}, v_j^{(k)})$ should be added into the corresponding point set P_j. When the known process is over, for each class j, there exists one point set:

$$P_j = \{(u_j^{(k_1)}, v_j^{(k_1)}), (u_j^{(k_2)}, v_j^{(k_2)}), ..., (u_j^{(k_{n_j})}, v_j^{(k_{n_j})})\},$$

where n_j is the number of the samples which are correctly classified as class j in the known class detection phase and there are totally N+1 point sets.

One-Class SVM method [22] is implemented to fit one decision function $f_j(\boldsymbol{p})$ for each point set P_j. The sign of the decision function shows whether the input is an outlier. When classifying one testing sample $\boldsymbol{x}$, we should get the initial class prediction j from the output of the classification model and extract the corresponding point (u_j, v_j) in advance as Figure 2. If the decision function $f_j(\boldsymbol{p}) < 0$, p can be predicated as an outlier, this testing sample x can be predicated as a sample from unknown classes of domain names. In DGA domain name detection task, this input domain should be recognized as an unknown class of DGA domain name and network security experts will further analyze it. In the other condition, if $f_j(\boldsymbol{p}) \geq 0$, it should still be predicted as a domain name from known class j.

The OpenMax model can output a probability vector $\boldsymbol{v'} \in \mathbb{R}^{(N+2)\times 1}$ to show the probability that one input belongs to a known

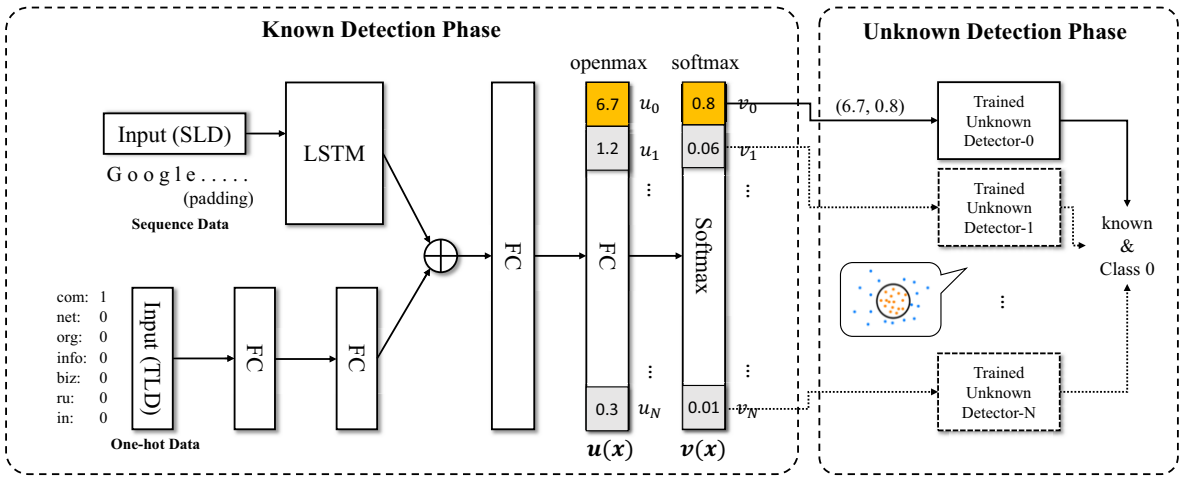


Figure 2. Overall Architecture of OpenSMax Model

Algorithm 1 Generate One-Class SVM models for each known class

Input: Activation output for softmax layer $\boldsymbol{v}(\boldsymbol{x^{(i)}}) \in \mathbb{R}^{(N+1)\times 1}$ and penultimate layer $\boldsymbol{u}(\boldsymbol{x^{(i)}}) \in \mathbb{R}^{(N+1)\times 1}$ for each training data $i, i \in [1, M]$, where M is number of the training samples.

Output: unknown class decision function for each class j: $f_j(\boldsymbol{p}), j \in [0, N]$

for each known class $j = 0$ to N **do**
  $P_j = \varnothing$
end for
for each training sample $i = 1$ to M **do**
  $j =$ the prediction result of sample $\boldsymbol{x}^{(i)}$
  $P_j = P_j \bigcup (u_j(\boldsymbol{x}^{(i)}), v_j(\boldsymbol{x}^{(i)}))$
end for
for each known class $j = 0$ to N **do**
  Use the set P_j to build a one-class SVM decision function $f_j(\boldsymbol{p})$, $\boldsymbol{p} \in \mathbb{R}^2$
end for

class (from 0 to N) or an unknown class (N+1). Similarly, in OpenSMax, the decision function of the One-Class SVM can also be used to give a probability vector. The process can be seen in Algorithm 2. The maximum activation value minus the output of the decision function is defined as unknown activation value (Eq. 1). When the decision function of the One-Class SVM takes a point as the input, the function will output a signed distance from this point to the boundary of the decision hyper-plane. Under this circumstance, the larger the value of the distance is, the more likely the point is from an unknown class. After that, unknown activation value is used to update probability vector by softmax function (Eq. 2). In next section, we can formally prove this method can manage open space risk.

4.4 Mean Value Restrictive Condition

When we use One-Class SVM model to calculate the decision boundary, we find that not all outliers belong to unknown classes of DGA domain names in our experiment. E.g., both $u_j(\boldsymbol{x})$ and $v_j(\boldsymbol{x})$ are very large values, but it does not mean the domain name $\boldsymbol{x}$ belongs to an unknown class. On the contrary, it is most likely to be a sample from a known class because of the high confidence value.

Algorithm 2 Calculate unknown/known class probability vector in OpenSMax

Input: Prediction data $\boldsymbol{x}$, Prediction class from Classification Model j_0, unknown class decision function for class j_0: $f_{j_0}(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x}))$ and Activation vector for penultimate layer $\boldsymbol{u}(\boldsymbol{x}) \in \mathbb{R}^{(N+1)\times 1}$

Output: The probability vector $\boldsymbol{v'} \in \mathbb{R}^{(N+2)\times 1}$ of input data $\boldsymbol{x}$

for each class $j = 0$ to N **do**
  $u'_j = u_j(\boldsymbol{x})$
end for

$$u'_{N+1} = u_{j_0}(\boldsymbol{x}) - f_{j_0}(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x})) \quad (1)$$

for each class $j = 0$ to $N + 1$ **do**

$$v'_j = \frac{e^{u'_j}}{\sum_{i=0}^{N+1} e^{u'_i}} \quad (2)$$

end for

Therefore, we introduce two restrictive conditions when we make a classification decision. First, we need to calculate mean value from each point set P_j:

$$(\bar{u}_j, \bar{v}_j) = \frac{1}{n_j}(\sum_{i=1}^{n_j} u_j^{(i)}, \sum_{i=1}^{n_j} v_j^{(i)}), n_j = |P_j|$$

For one input data $\boldsymbol{x}$, only when it satisfies all of the following restrictive conditions, we can predict it as an unknown class:

- $f_{j_0}(\,(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x}))\,) < 0$
- $u_{j_0}(\boldsymbol{x}) < \bar{u}_{j_0}$
- $v_{j_0}(\boldsymbol{x}) < \bar{v}_{j_0}$

Otherwise, we still keep the initial prediction label. These conditions are checked at the last step of the unknown class detection phase (just before outputting results). We call the last two restrictive conditions Mean Value Restrictive Conditions (MVRCs).

5 OPENSMAX COMPACT ABATING PROPERTY PROOF

According to the theory research in open set recognition issue, if an open set recognition method can build a function which satisfies the Compact Abating Property (CAP), this method can manage open space risk [16]. In other words, models which satisfy this property are more convincing to use in the open set recognition issue. Open space risk is formally defined as Defination 1. It is a function to relatively measure the probability of labeling the samples of unknown classes (in the open space) as known classes.

Definition 1 *(Open space risk) Let g be a measurable recognition function where $g(\boldsymbol{x}) \in [0,1]$, demonstrating the probability that the sample $\boldsymbol{x}$ belongs to a known class. $\mathcal{O}$ represents the open set space (excluding the known space), and S_o represents the full space including open space and known space. Then Open Space Risk $R_{\mathcal{O}}(g)$ can be defined as*

$$R_O(g) = \frac{\int_{\mathcal{O}} g(x)dx}{\int_{S_o} g(x)dx}$$

The more we label the samples in open space (unknown space) as known classes, the greater the open space risk is. Therefore, we need to narrow the space where we tend to label the samples as known classes. [16] points out that if a recognition function is a function which monotonically decreases with distance between the input sample and the known class data in feature space. It has been proved that setting a threshold for this function can limit the open space risk. Experts define the Compact Abating Property (CAP) in definition 2.

Definition 2 *(Compact Abating Property function) $A(r) : \mathbb{R} \to \mathbb{R}$ is a non-negative finite square integrable continuous decreasing function. When $\forall \boldsymbol{x}, \exists \boldsymbol{x}^*, s.t.\, g(\boldsymbol{x}) \leq A(\|\boldsymbol{x} - \boldsymbol{x}^*\|)$, we call $g(\boldsymbol{x})$ an abating function. The model with abating function can be deemed as a CAP model.*

Scheirer et al. [16] prove CAP model can largely limit the Open Space Risk. So in this section, we will prove OpenSMax model is also a CAP model, and hence, it is an reasonable open set recognition method. First, we need to define the feature space of input $\boldsymbol{x}$. As the former subsection describes, for input data $\boldsymbol{x}$,its feature space is $(\|\boldsymbol{u}(\boldsymbol{x})\|_\infty, \|\boldsymbol{v}(\boldsymbol{x})\|_\infty) = (u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))$. It can be considered as a non-linear transformation of the origin features. Our recognition function is $g(\boldsymbol{x}) = u'_{N+1}$, while u'_{N+1} is in Eq. 2. Also, according to Theorem 2 in [16], One-Class SVM can yield a CAP decision function. So, we have that all the $f_j(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))$ are CAP functions. Now we just need to prove the $g(\boldsymbol{x})$ is a CAP function.

Theorem 1 *The decision function $g(\boldsymbol{x})$ of OpenSMax model is a CAP function, and thus OpenSMax is a CAP model.*

Proof 1 *Since $f_j(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))$ is CAP function. According to Definition 2, we have non-negative finite square integrable continuous decreasing function $A(r) : \mathbb{R} \to \mathbb{R}$.*

$$\begin{aligned} u'_{N+1} &= \frac{e^{u'_{N+1}}}{\sum_{i=1}^{N+1} e^{u'_i}} \\ &= \frac{e^{u'_{j_0} - f_{j_0}(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))}}{\sum_{i=1, i\neq j_0}^{N} e^{u'_i} + e^{u'_{j_0}} + e^{u'_{j_0} - f_{j_0}(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x}))}} \\ &= \frac{e^{-f_{j_0}(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x}))}}{\frac{\sum_{i=1, i\neq j_0}^{N} e^{u'_i}}{e^{u'_{j_0}}} + 1 + e^{-f_{j_0}(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))}} \end{aligned}$$

In our 2D feature space, when $\| \boldsymbol{x} - \boldsymbol{x}^* \| \leq r$, we can have $u'_{j_0}(\boldsymbol{x}) \leq u'_{j_0}(\boldsymbol{x}^*) + r$. So, $e^{u'_{j_0}} = e^{u'_{j_0}(\boldsymbol{x})} \leq e^{u'_{j_0}(\boldsymbol{x}^*)+r}$. As a result, we have

$$u'_{N+1} \leq \frac{e^{-f_{j_0}(u_{j_0}(\boldsymbol{x}), v_{j_0}(\boldsymbol{x}))}}{\frac{\sum_{i=1, i\neq j_0}^{N} e^{u'_i}}{e^{u'_{j_0}(\boldsymbol{x}^*)+r}} + 1 + e^{-f_{j_0}(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))}}$$

Also, we have decreasing function $A_0(x^*)$, s.t.

$$-f_{j_0}(u_j(\boldsymbol{x}), v_j(\boldsymbol{x})) \leq A_0(\| \boldsymbol{x} - \boldsymbol{x}^* \|)$$

.

So, we have

$$u'_{N+1} \leq \frac{e^{A_0(\|\boldsymbol{x}-\boldsymbol{x}^*\|)}}{\frac{\sum_{i=0, i\neq j_0}^{N} e^{u'_i}}{e^{u'_{j_0}(\boldsymbol{x}^*)+r}} + 1 + e^{A_0(\|\boldsymbol{x}-\boldsymbol{x}^*\|)}}$$

And we can let

$$A_1(\|\boldsymbol{x} - \boldsymbol{x}^*\|) = \frac{e^{A_0(\|\boldsymbol{x}-\boldsymbol{x}^*\|)}}{\frac{\sum_{i=0, i\neq j_0}^{N} e^{u'_i}}{e^{u'_{j_0}(\boldsymbol{x}^*)+r}} + 1 + e^{A_0(\|\boldsymbol{x}-\boldsymbol{x}^*\|)}}$$

Due to the $A_0(r)$ is a non-negative finite square integrable continuous decreasing function, we can easily prove the $A_1(r)$ has the same characteristics. Then, we have $\forall \boldsymbol{x}, \exists \boldsymbol{x}^*, s.t.\, g(\boldsymbol{x}) = u'_{N+1}(\boldsymbol{x}) \leq A_1(\|\boldsymbol{x} - \boldsymbol{x}^*\|)$ So,it satisfies the definition of the CAP function and the open space risk is limited.

6 EVALUATION

6.1 Dataset & Data Preprocessing

For benign domain names, we choose to use the generic benign domain name dataset Alexa-1M [23], which is a list of top website. For DGA domain names, we use the DGA datasets which is collected by DGA Detecting System of 360 Company [3]. It contains 43 classes of DGA domain names in total and these DGA domain names have different characteristics.

We select 21 classes of DGA domain names with more than 1,000 samples as Table 1. Also, we randomly extract 4,000 samples if one class of DGA domain names has more than 4,000 ones in order to keep the balance between these classes. We always use label 0 to represent the benign domain names, and use other labels to represent DGA domain names. The maximum number label represents the unknown classes of DGA domain names. E.g., if we use eleven classes of DGA domain names and one class of benign domain names to build the training set, we will use 0 to represent the benign class, $\{1, 2, 3, ..., 11\}$ to represent corresponding known classes of DGA domain names and 12 to represent the rest (unknown classes) of DGA domain names. The corresponding labels for classes of DGA domain names can be found in Table 1.

6.2 Baseline Models

We compare our proposed model with two well-known open set recognition methods - OpenMax and DOC, which are proposed for computer vision and text classification. We do not implement G-OpenMax and CROSR because they are not suitable for text like unknown class detection issue.

[3] http://data.netlab.360.com/dga/

Table 1. Selected DGA Classes for Evaluation

No.	DGA Class	No.	DGA class	No.	DGA Class
1	banjori	8	murofet	15	shifu
2	chinad	9	necurs	16	shiotob
3	cryptolocker	10	pykspa_v1	17	simda
4	dyre	11	qadars	18	suppobox
5	emotet	12	ramnit	19	symmi
6	gameover	13	ranbyus	20	tinba
7	locky	14	rovnix	21	virut

OpenMax model is proposed for open set recognition in computer vision [11]. It is a CNN-based model, and it extracts the openmax layer as the feature space. It implements the Extreme Value Theory (EVT) and build an EVT recognition function for each class. In our experiment , we extract the openmax layer of known DGA domain name detection model and we find the best result by making grid search across the hyperparameter values including the number of top classes to revise α and Weibull tail size.

DOC method [18] is also a CNN-based model for text classification. It replaces the softmax layer with the sigmoid layer and this method uses the output of the sigmoid layer to fit a Gaussian distribution for each class. It detects unknown class by using the parameters of the corresponding Gaussian distribution and we get the best result by adjusting hyperparameter values.

For OpenMax model, we implement it based on the published OpenMax code[4]. For Doc model, we re-implement it by ourselves with deep learning library Keras.

6.3 OpenSMax Models

To make a more comprehensive comparison with OpenMax method, we extract different layers and elements to build different One-Class SVM models in unknown class detection phase, which are listed in Table 3. We use the notation OpenSMax-*x*-*y*D to name them, where *x* is the number of the layers we use for outlier detection and *y* is the number of the elements we extract from the activation vector of each layer to build the One-Class SVM models.

In our experiments, we extract the maximum activation values of the softmax layer and openmax layer to build One-Class SVM models. According the naming rule, we call it OpenSMax-2-1D model. Similarly, we implement a method OpenSMax-1-1D which only extracts the maximum activation value of the openmax layer to build One-Class SVM models. In addition, we try to extract all the elements of activation vector. We call them OpenSMax-2-(N+1)D model and OpenSMax-1-(N+1)D model, because these extracted activation vectors are (N+1) dimensions.

Table 2. Classification Model results for DGA Detection

Model	Acc	Acc (50% unknown)
Bigram (SLD)	80.27	57.62
LSTM (SLD)	90.03	64.63
Bigram (SLD) + LSTM (SLD)	90.23	64.77
LSTM (SLD + TLD)	95.51	69.06
LSTM (SLD) + One-hot (TLD)	**96.73**	**69.34**

[4] https://github.com/abhijitbendale/OSDN

6.4 Known Class Detection Phase Evaluation

In known class detection phase evaluation, we test different models for DGA domain name detection, including Bigram [24, 3], LSTM [6] with SLD. Then, we test the performance of the model which combines Bigram features and LSTM features with SLD for reference. In addition, we test the LSTM model by inputting the whole domain (SLD+TLD). In ordinary dataset, both the training set and the testing set contains all 21 classes of DGA domain names and 1 benign domain class. Also, as introduced in the section before, we randomly select 11 DGA classes as unknown classes and these classes only appear in the testing set as 50% unknown classes. The evaluation results are shown in Table 2.

From experiment results, our model using one-hot encoding to represent TLD achieves the highest accuracy 96.73% among these models. Also, we can find that the accuracy of our model is 1.22% higher than using LSTM model directly. The experiment results indicate that Top-Level Domain can provide valuable information for DGA detection and it can help us significantly increase detection accuracy.

6.5 Unknown Class Detection Phase Evaluation

According to experimental methods from most papers [18, 20], we use 25%, 50%, 75% of DGA classes as known classes of DGA domain names in the training set. For data balance, we randomly select 3,000 domain names from unknown classes of DGA domain names.

If the class of DGA domain name does not appear in the training set, we label it as 'unknown'. Otherwise, we label it as its initial class. Therefore, we view it as a normal multi-classification task and the number of classes is N+1. In addition, it should be noted that the label 'N+1' does not appear in the training set.

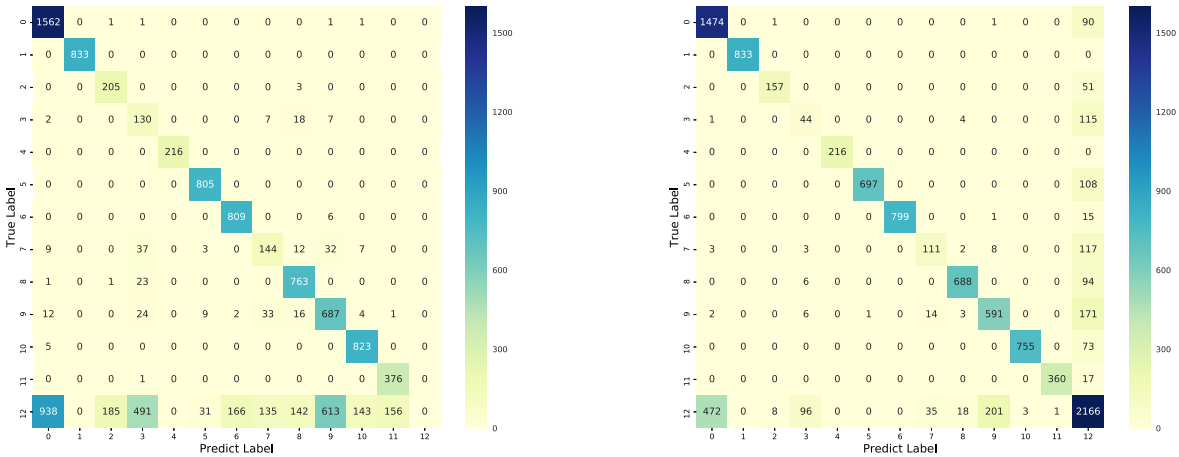
For the experiment results, we use both accuracy and Macro F1-Score as evaluation metrics. Accuracy means the ratio between the number of the samples which are classified correctly and the number of the total samples in the testing set. To calculate Macro F1-Score, firstly we need to calculate F1-Score (the Harmonic average of precision and recall) for each class separately. Macro F1-Score is the arithmetic mean value of F1 Score of each class [25]. Macro F1-Score considers each class's recognition result equally. As a result, it is widely used in the evaluation of open set recognition.

For each OpenSMax model, we find the best accuracy and Macro F1-Score by adjusting their hyperparameters. Experiment results after grid search of hyperparameters can be seen in Table 4. The effect of OpenSMax model can be seen in the confusion matrix as Figure 3. In Figure 3(a), we can see that without the unknown detector, we must classify an input into a known class, and consequently make many mistakes for domain names from unknown DGA classes. 31.3% of domain names of unknown DGA classes are recognized as benign domain (label 0). When we add our unknown detection method (OpenSMax), we can find that 72.2% of unknown DGA domain names can be classified correctly in Figure 3(b).

In these results, we can find that our OpenSMax-2-1D and OpenSMax-2-1D+MVRC model achieves the highest accuracy 83.46% and 83.62% in 50% unknown class dataset, which performs better than DOC and OpenMax models. The reason is that OpenMax method uses the full vector of the openmax layer as the feature space, but most of values in the vector are small and become noise in unknown class detection. DOC only uses the maximum value last sigmoid layer to fix Gauss distribution, but does not take the relation between other values and maximum value into account. In our exper-

Table 3. OpenSMax Related Models. The second and third column are layers and elements we extract to build the outlier detection model. The fourth column is mathematical notation for extracted data.

Model Name	Layer(s)	Max/All layer element(s)	Mathematics Symbol of Input
OpenSMax-1-1D	openmax	Max	$u_j(\boldsymbol{x})$
OpenSMax-1-(N+1)D	openmax	All	$\boldsymbol{u}(\boldsymbol{x})$
OpenSMax-2-(N+1)D	openmax+softmax	All	$(\boldsymbol{u}(\boldsymbol{x}), \boldsymbol{v}(\boldsymbol{x}))$
OpenSMax-2-1D	openmax+softmax	Max	$(u_j(\boldsymbol{x}), v_j(\boldsymbol{x}))$



(a) CM (Before Unknown Detection, Acc = 69.16%)

(b) CM (After Unknown Detection, Acc = 83.62%)

Figure 3. Effect of Proposed Method (OpenSMax-2-1D) in Confusion Matrix. The maximum label 12 represents the unknown DGA class. The minimum label 0 represents the benign domain class. The classes which labels 1 to 11 represent are shown in Table 1.

iment, we use both the maximum value of the openmax layer and the softmax layer to achieve the best results.

When introducing the mean value restrictive conditions, we only consider the value lower than the mean value for one class. So, we can prevent to classify the sample with high confidence into unknown class. From the results, we can find that this improving method preforms better in conditions where more classes of DGA domain names are known because the confidence accuracy is more accurate when the model gets more information from different classes.

Also, in our OpenSMax related models, our method OpenSMax-2-1D performs better than other OpenSMax models, which is 1.48% higher than the second place OpenSMax-1-1D. From the comparison of OpenSMax-1-1D and OpenSMax-2-1D, we can find that the softmax layer can provide the model with useful information. From the comparison of OpenSMax-2-1D and OpenSMax-2-(N+1)D, we can find that other elements in layer vectors can provide little information and can become noise in outlier detection.

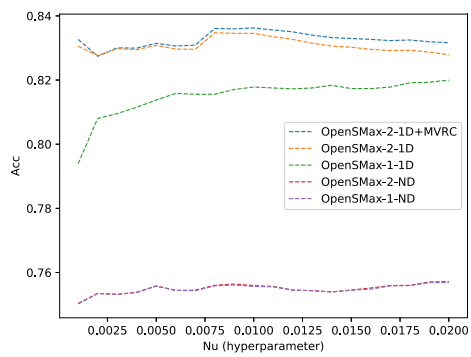


Figure 4. Hyperparameter Search for OpenSMax Related Models

6.6 Hyperparameter Setting

The hyperparameter of OpenMax includes the tail size for EVT calibration and α, the number of top classes to revision. Tail size largely depends on the number of the samples from each class. Also, α depends on the number of total classes. Given that our dataset is rather different from the dataset ILSVRC used in the origin paper, we use grid search to find the best hyperparameter values rather than the value used in paper. Refering to our dataset, the range of the tail size is set to [50, 500), step=50 and α is set to [4,8), step=1. In total 9×4 =36 results, the best accuracy 77.62% appears in (200,6) and the best Macro F1-Score 78.85% appears in (100,6).

The main hyperparameter of DOC is the weight α which means if the value of one sample is α times of standard deviations away from the mean value, it is considered as an outlier. We set α to [10,100], step=10, and find the best result. We get the highest accuracy 74.12% at α=20 and the highest Macro F1-Score 70.20% at α=90.

For OpenSMax related models, the main hyperparameter is nu,

Table 4. Accuracy and Macro-F1 Score for different methods

Method	25%		50%		75%	
	Acc	Macro F1	Acc	Macro F1	Acc	Macro F1
DOC[18]	62.08	55.19	74.12	70.20	72.56	77.16
OpenMax[11]	84.39	81.06	77.62	78.85	75.35	81.05
OpenSMax-1-(N+1)D	82.59	78.48	77.67	75.56	75.37	81.19
OpenSMax-2-(N+1)D	82.59	78.48	75.59	77.69	75.36	81.18
OpenSMax-1-1D	84.73	81.34	81.78	81.90	80.36	84.50
OpenSMax-2-1D	**85.29**	**82.65**	83.46	**83.24**	80.67	85.04
OpenSMax-2-1D+MVRC	85.12	82.31	**83.62**	83.23	**80.78**	**85.10**

and it can control the size of decision boundary. It is an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. The larger the nu is, the smaller the known decision space is, which means we are more likely to recognize a sample as an unknown class. Figure 4 shows the process of adjusting nu to find the best accuracy result with 50% unknown classes. When nu=0.010, our proposed method can reach the highest accuracy.

7 CONCLUSION

This paper proposes the OpenSMax method for DGA domain name detection which can be used in the open domain name dataset. Not only can it detect known classes of DGA domain names, but also it can also discover unknown classes of DGA domain names. We have proved this method has limited open space risk. Our OpenSMax method performs better than the start-of-art open set methods which have been used in computer vision and text classification. Also, we propose a possible improving method for avoiding samples with high confidence being classified as the unknown class. In the future, we will use more real datasets in cyberspace to verify our method.

ACKNOWLEDGEMENTS

This work is supported by National Key R&D Program of China (No. 2019QY1402).

REFERENCES

[1] Congyuan Xu, Jizhong Shen, and Xin Du. Detection method of domain names generated by dgas based on semantic representation and deep neural network. *Computers & Security*, 85:77–88, 2019.

[2] Daniel Plohmann, Khaled Yakdan, Michael Klatt, Johannes Bader, and Elmar Gerhards-Padilla. A comprehensive measurement study of domain generating malware. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 263–278, 2016.

[3] Sandeep Yadav, Ashwath Kumar Krishna Reddy, AL Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated domain-flux attacks with dns traffic analysis. *IEEE/Acm Transactions on Networking*, 20(5):1663–1677, 2012.

[4] Tzy-Shiah Wang, Hui-Tang Lin, Wei-Tsung Cheng, and Chang-Yu Chen. Dbod: Clustering and detecting dga-based botnets using dns traffic analysis. *Computers & Security*, 64:1–15, 2017.

[5] Manos Antonakakis, Roberto Perdisci, Yacin Nadji, Nikolaos Vasiloglou, Saeed Abu-Nimeh, Wenke Lee, and David Dagon. From throw-away traffic to bots: detecting the rise of dga-based malware. In *Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12)*, pages 491–506, 2012.

[6] Jonathan Woodbridge, Hyrum S Anderson, Anjum Ahuja, and Daniel Grant. Predicting domain generation algorithms with long short-term memory networks. *arXiv preprint arXiv:1611.00791*, 2016.

[7] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1893–1902, 2015.

[8] Zheng Wang, Xiaojun Ye, Chaokun Wang, YueXin Wu, Changping Wang, and Kaiwen Liang. RSDNE: Exploring relaxed similarity and dissimilarity from completely-imbalanced labels for network embedding. In *AAAI*, pages 475–482, 2018.

[9] Zheng Wang, Xiaojun Ye, Chaokun Wang, Jian Cui, and Philip S Yu. Network embedding with completely-imbalanced labels. *TKDE*, 2020.

[10] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *arXiv preprint arXiv:1811.08581*, 2018.

[11] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572, 2016.

[12] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772, 2012.

[13] Miranda Mowbray and Josiah Hagen. Finding domain-generation algorithms by looking at length distribution. In *2014 IEEE international symposium on software reliability engineering workshops*, pages 395–400. IEEE, 2014.

[14] Wen-Jie Song and Bin Li. A method to detect machine generated domain names based on random forest algorithm. In *2016 International Conference on Information System and Artificial Intelligence (ISAI)*, pages 509–513. IEEE, 2016.

[15] 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*, pages 211–218. ACM, 2017.

[16] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2317–2324, 2014.

[17] David MJ Tax and Robert PW Duin. Support vector domain description. *Pattern recognition letters*, 20(11-13):1191–1199, 1999.

[18] Lei Shu, Hu Xu, and Bing Liu. Doc: Deep open classification of text documents. *arXiv preprint arXiv:1709.08716*, 2017.

[19] ZongYuan Ge, Sergey Demyanov, Zetao Chen, and Rahil Garnavi. Generative openmax for multi-class open set classification. *arXiv preprint arXiv:1707.07418*, 2017.

[20] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Classification-reconstruction learning for open-set recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4016–4025, 2019.

[21] Bin Yu, Jie Pan, Jiaming Hu, Anderson Nascimento, and Martine De Cock. Character level based detection of dga domain names. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[22] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588, 2000.

[23] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1388–1401. ACM, 2016.

[24] Sandeep Yadav, Ashwath Kumar Krishna Reddy, AL Reddy, and Supranamaya Ranjan. Detecting algorithmically generated malicious domain names. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 48–61. ACM, 2010.

[25] Vincent Van Asch. Macro-and micro-averaged evaluation measures. *Tech. Rep.*, 2013.