

# Malicious Domain Detection Based on K-means and SMOTE

Qing Wang<sup>1,2</sup>, Linyu Li<sup>1,2</sup>, Bo Jiang<sup>1,\*</sup>, Zhigang Lu<sup>1,2</sup>, Junrong Liu<sup>1</sup>, and Shijie Jian<sup>1,2</sup>

<sup>1</sup> Institute of Information Engineering, Chinese Academy of Sciences,  
Beijing, China 100093

<sup>2</sup> School of Cyber Security, University of Chinese Academy of Sciences,  
Beijing, China 100029

{wangqing, lilinyu, jiangbo, luzhigang, liujunrong, jianshijie}@iie.ac.cn

**Abstract.** The Domain Name System (DNS) as the foundation of Internet, has been widely used by cybercriminals. A lot of malicious domain detection methods have received significant success in the past decades. However, existing detection methods usually use classification-based and association-based representations, which are not capable of dealing with the imbalanced problem between malicious and benign domains. To solve the problem, we propose a novel domain detection system named KSDom. KSDom designs a data collector to collect a large number of DNS traffic data and rich external DNS-related data, then employs K-means and SMOTE method to handle the imbalanced data. Finally, KSDom uses Categorical Boosting (CatBoost) algorithm to identify malicious domains. Comprehensive experimental results clearly show the effectiveness of our KSDom system and prove its good robustness in imbalanced datasets with different ratios. KSDom still has high accuracy even in extremely imbalanced DNS traffic.

**Keywords:** Malware Domain Detection · Data imbalance · K-means · SMOTE · CatBoost

## 1 Introduction

On the Internet, malicious activities are everywhere. As the basis of the Internet, the Domain Name System implements domain-to-IP address mapping, now is widely used by cybercriminals for malicious activities. Attackers use malicious domains for phishing, distributing malware, controlling botnets and other malicious activities. Effective detection of malicious domain names on the Internet is critical.

Malicious activities leave traces in DNS traffic, so DNS traffic can be used to detect malicious domains. Traditional detection systems analyse various DNS traffic and extract features from DNS traffic which can effectively distinguish between malicious and benign domains, such as the number of domain resolution

---

\* Corresponding author

IP addresses [1], the standard deviation of TTL (Time-To-Live) values [2], then build a classifier through these features. This classifier can be used to detect unlabeled domains after being trained by labeled datasets.

Some researchers use an association-based approach to detect malicious domains. This method is based on the intuition that the domain associated with a malicious domain is likely to be malicious as well. By establishing association rules between domains, DNS-related data is modeled as domain-IP bipartite graphs [3] or domain-client bipartite graphs [4] or more complexity graphs [5], which is constructed to discover more malicious domains.

These existing methods have achieved excellent results to a certain extent, but rarely consider the problem of data imbalance. In the real network environment, the number of malicious domains is much smaller than benign domains, and the DNS traffic between benign domains and malicious domains is extremely imbalanced. Imbalanced datasets have a significant impact on the learning of the classifier and affect the effect of actual use.

In order to solve this problem, we propose a malicious domains detection system named KSDom, which can effectively detect malicious domains in imbalanced DNS traffic and has significant performance.

In this paper, we make the following contributions:

- We integrate active DNS data with rich external data, select powerful features that can be used to distinguish malicious domains to build a detection system based on CatBoost algorithm, which can effectively classify malicious domains.
- We propose a malicious domain detection system named KSDom based on CatBoost algorithm, which can detect malicious domains from the imbalanced DNS traffic. KSDom uses K-means and SMOTE method to handle the imbalanced DNS traffic, then uses our classifier based on CatBoost algorithm to detect malicious domains.
- The experimental results demonstrate our system is effective, competitive and can achieve state-of-the-art performance.

The rest of this paper is organized as follows. We present related work in Section 2 and describe our system framework and each component of it in Section 3. Section 4 presents the experimental results and the results of comparison experiments of several models. Section 5 presents the discussion.

## 2 Related Work

### 2.1 Malicious Domain Detection

The detection methods in the field of malicious domains can mainly be divided into two categories: classification-based methods and association-based methods. In the classification-based methods, [1] established a reputation system that dynamically assigns reputation scores to domains based on the features of the network and DNS zones and assigns low reputation scores if the domain name

involves malicious activity. [6] presented a detection technology for detecting DGA domains, which is based on the following principle: domains of the same DGA algorithm will generate similar Non-Existing Domains (NXDomain) traffic. The technology uses a combination of aggregation and classification algorithms to cluster NXDomain responses to detect the relevant DGA domains. [7] used a combination of IP and string-based features to detect DGA-generated domains. [2] extracted fifteen features based on passive DNS traffic to detect different types of malicious domains, not just detects a particular type of domain. [8] used only time-of-registration features to identify malware domains when they are registered. [9] used passive DNS traffic to record domain name query history in the real network environment to generate eighteen features to detect Fast-Flux domains.

In the association-based method, systems use the association between domains and domain-related IPs (IP address) or clients to form domain graphs, then execute the inference algorithm on the domain graphs to discover more malicious domains. [10] built a client-domain bipartite graph based on the clients and the domain which it queries, effectively tracking the malware-controlled domains in the ISP network and discovering new malware control domains. [4] formed the domain-IP bipartite graph through the association between domains and IPs, then used a path-based algorithm to discover potential malicious domains. The above systems all use bipartite graphs, which only represents two types of nodes and one type of relationship. [5] constructed graphs by using multiple types of nodes and relationship and the client, domain, IP address and their relationship are modeled as HIN model. The similarity of the combined domains created on multiple meta paths represents the information contained in the DNS data.

## 2.2 Imbalanced Data

In recent years, more and more research begins to pay attention to the importance of imbalanced data and apply it to various fields [11], such as fraud detection, text classification and medical diagnosis in banks [12]. The extremely imbalanced data leads to poor performance of the trained classifier.

In general, there are two ways to solve the imbalanced data problem. The first one is from the algorithm layer, by modifying the bias of the algorithm on the datasets, so that the decision plane can tend to minority class samples, improving the recognition rate of minority class samples. This method is based on cost-sensitive learning [13], and the representing algorithm is AdaCost. The second method is from the data layer, by resampling imbalanced data. Resampling includes two methods: undersampling and oversampling. Undersampling, that is, by extracting some samples from majority class samples, but this may lead to the loss of classification information. The improved undersampling method is EasyEnsemble [14] and BalanceCascade [15]. Oversampling, that is, by increasing the number of samples of minority class samples, the original data can be well preserved, but it is easy to produce overfitting. The improved oversampling algorithm includes SMOTE (Synthetic Minority Oversampling Technique) [16].

There are also some hybrid methods. [17] proposed a novel hybrid learning framework which combines data-level and algorithm-level methods to deal with data imbalanced problem. [18] combined oversampling and undersampling techniques and SVM approach to handle the imbalanced data classification problem.

Unlike algorithm layer methods that need to be combined with specific classifiers, data layer methods are universally applicable, so most research methods are aimed at considering the processing of imbalanced datasets from a data perspective. In terms of network security, [19] combined ensemble learning with undersampling, using an improved EasyEnsemble method to learn imbalanced DNS traffic data and generate a classification model which can detect malicious domains. Illuminated by this research which shows resampling usefulness in malicious domain detection.

The classification model generated by using ensemble learning combined with undersampling is prone to noise, and the Decision Tree algorithm ignores the correlation between features, resulting in poor model performance. We propose KSMOTE method, which can deal with imbalanced datasets and avoid noise well and combine with the CatBoost algorithm, which can deal with the correlation between features and avoid data overfitting.

### 3 Proposed KSDom System

The goal of KSDom is to detect malicious domains on the Internet. KSDom is based on the following intuitions: (i) when malicious domains perform malicious activities, the generated DNS traffic is different with the DNS traffic generated by benign domains, and attackers have to consider the costs of network resources, they prefer to choose domains with lower registration costs and tend to reuse network resources. (ii) In the real Internet, the number of benign domains is much bigger than malicious domains, DNS traffic is extremely imbalanced between benign domains and malicious domains, the classifiers trained by balanced datasets have poor performance in the real network.

Therefore, our system collects a large number of DNS traffic on the Internet, analyzes DNS traffic and selects useful features from it, oversamples the imbalanced training samples, and trains the processed samples to generate a malicious domain detection model to detect unlabeled domains. As shown in Figure 1, KSDom consists of four main components: Data Collector, Feature Selection, Imbalanced Data Processor and Classifier component.

The operational workflow of KSDom is as follows. Data Collector aggregates imbalanced Active DNS traffic data from the local network and expands rich DNS-related data (step 1), Feature Selection component selects features from DNS traffic data to generate training samples (step 2), put the imbalanced training samples into the third component: Imbalanced Data Processor, processes imbalanced training samples with our processor (step 3), the output is fed into the fourth component: Classifier. This module trains labeled datasets based on CatBoost algorithm to build a malicious domain detection classifier (step 4),

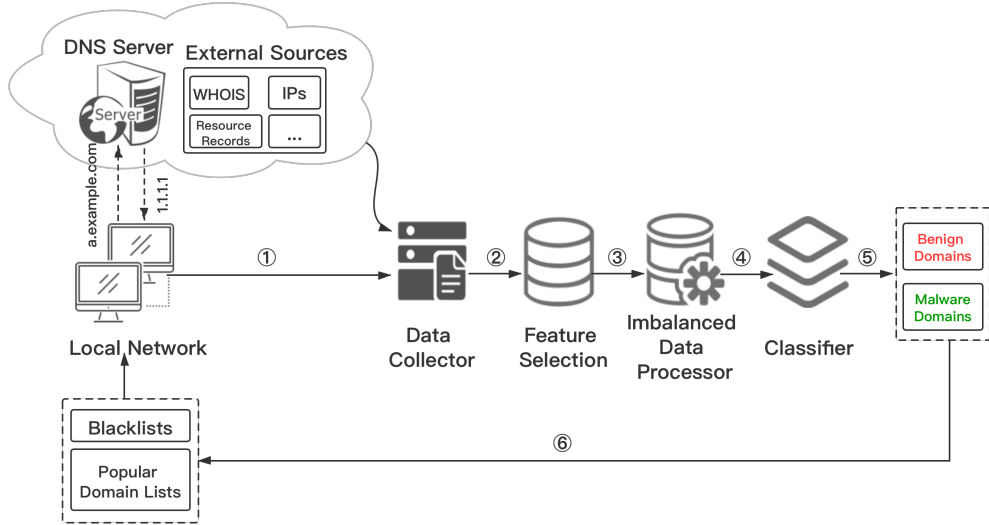


Fig. 1: Overview of KSDom

the generated classifier categorizes unlabeled domains (step 5), we analyze the classification results and add labeled domains to our domain list (step 6).

In the following, we will introduce each component in detail.

### 3.1 Data Collector

In the Data Collector module, we first describe the type of DNS traffic data. We divide DNS traffic into Active DNS traffic data and Passive DNS traffic data according to the collection method.

Active DNS data refers to DNS data obtained by periodically proactively parsing a large number of domains [20], which are collected from multiple public sources, such as the Alexa Top Sites [21], public whitelists, various public blacklists, etc. Active DNS data mainly captures the DNS records of domains, such as the records of domain resolved IP addresses, the information of name servers, etc. Active DNS data does not have the information of users query domains, so it does not involve any privacy issues.

Passive DNS data is obtained by deploying sensors on the DNS servers or accessing the DNS server logs to obtain real DNS queries and response information. Passive DNS data provides a summarized view of domain queries. Experiments have shown that active DNS data provides more kinds of records, and passive DNS data provides a tighter connection graph [22]. Passive DNS can provide richer information than active DNS, but due to privacy issues and the location of the deployed sensors, the collected data has certain limitations.

Compare to passive DNS data, the collection method of active DNS is flexible, and data collector can easily query domains of the collection lists without any

private issues, and active DNS data can discover potentially malicious domains that are newly registered but not yet used [20].

In this research, we use active DNS traffic data and DNS-related external data for data enrichment. Our data collector collects the following data:

- Active DNS traffic data. We use the local network to actively query the domains. After the DNS server receives the query requests, it returns the corresponding response results. We collect response data as input to the data collector.
- External data related to DNS. We combine collected active DNS traffic data with DNS-related external data to enrich our datasets, such as the registration records of domains, the WHOIS information, the public IP blacklists information and other information from the associated Resource Records (RRs).

### 3.2 Feature Selection

This component is used to process the features we select in the DNS traffic data. In order to select features that can effectively differentiate malicious and benign domains, large amounts of DNS traffic need to be observed. By analyzing and summarizing the large amount of data obtained, we extract 16 composite features to build malicious detection classifier, as shown in Table 1. We will describe these features and explain why they can distinguish malicious domains.

Table 1: Selected Domain Feature Set

| Feature Set               | ID | Feature Name  | Importance |
|---------------------------|----|---|------------|
| Domain Name-Based Feature | 1  | Length of domain names                                    | 0.01306    |
|                           | 2  | Distribution of vowel characters                          | 0.04002    |
|                           | 3  | Distribution of consonant characters                      | 0.17824    |
|                           | 4  | Conversion frequency of vowels and consonants             | 0.04263    |
|                           | 5  | Number of numeric characters                              | 0.00412    |
|                           | 6  | Ratio of numeric characters                               | 0.00561    |
|                           | 7  | Conversion frequency of numeric and alphabetic characters | 0.10680    |
|                           | 8  | Number of special characters                              | 0.04600    |
| DNS Answers-Based Feature | 9  | Number of distinct IP addresses                           | 0.06591    |
|                           | 10 | Number of MX records                                      | 0.09031    |
|                           | 11 | Number of domains that IP reversely resolves              | 0.06737    |
|                           | 12 | Number of distinct NSs                                    | 0.07227    |
|                           | 13 | Similarity of NS domain names                             | 0.03003    |
| Contextual Features       | 14 | Number of labels of the website's contents                | 0.11687    |
|                           | 15 | Number of malicious IP addresses                          | 0.19452    |
|                           | 16 | Validity period of a domain                               | 0.05746    |

DNS traffic features can be divided into internal features and contextual features. Internal features can be extracted directly from Resource Records, such as the record of domain resolution IP addresses, the character distribution of domain names. Contextual features are built from DNS-related external data information and associated Resource Records, such as Whois information for the domain, reverse mapping of domain resolved IP addresses, etc. These features can provide more information that may contain malicious behaviors than internal features, such as [23] found that domains associated with botnets typically do not have MX records.

We classify the features into three categories, based on their sources. (1) domain name-based features. These features are based on the domain name string itself, such as the character distribution of domain names. (2) DNS answer-based features. These features are based on DNS response data, such as Resource Records and IP information. (3) Contextual features. Contextual features come from external sources of data, such as WHOIS records and registration information.

**Domain Name-Based Features.** Benign domain applicants choose domain names that are easily remembered or beneficial to people’s pronunciation, and attackers often choose characters that are difficult to remember when constructing malicious domains. Through research on a large number of malicious domains, we find that malicious domains contain more numbers, and there are a large number of numbers and letters. Therefore, the malicious domain has a more confusing combination of numbers and words, and the proportion of consonants in the random malicious domain is larger, and the conversion frequency between the vowels and the consonants is higher. Therefore, the following characteristics are constructed: the length of domain name (Feature 1), the distribution of vowel characters (Feature 2) which consists of three atomic features: the number of vowels, the ratio of vowels and the number of vowel characters in the string without a domain name suffix. The distribution of consonant characters (Feature 3) as well as a composite feature which consists of 2 atomic features: the number of consonant characters and the number of consonants in the string without a domain name suffix. The conversion frequency of vowels and consonants (Feature 4), the number of numeric characters (Feature 5), the ratio of numeric characters (Feature 6), the conversion frequency of numeric and alphabetic characters (Feature 7), the number of special characters (Feature 8).

**DNS Answer-Based Features.** The DNS answer records of malicious domains are very different from that of benign domains. Malicious domains usually have more address records, because malicious domain holders usually choose domains that map many IP addresses, and malicious domains usually contain fewer MX (Mail Exchanger) records and more NS (Name Server) records. Based on this information, we choose the following characteristics: the number of distinct IP addresses (Feature 9), the number of MX records (Feature 10), the number of domains that IP reversely resolves (Feature 11), the number of distinct NSs (Feature 12), and the similarity of NS domain names (Feature 13).

**Contextual Features.** Attackers typically do not set the website contents for the malicious domains, or just copy the specific templates to fake the phishing websites. We extract the following feature: the number of labels of the website’s contents (Feature 14). Attackers usually reuse resources, and IP addresses associated with the malicious domains typically are also malicious. Query the public IP blacklists to gain how many of the IPs associated with a domain is malicious. We extract the following feature: the number of malicious IP addresses (Feature 15). Most malicious domains are used for short-lived malicious activities, and it will be soon to be recorded into blacklists, so registrants usually apply for malicious domains with shorter lifetimes. A malicious domain usually has a short validity period. The validity period of a domain can be determined by calculating the expiration date and registration date. We extract the following feature: the validity period of a domain (Feature 16).

### 3.3 Imbalance Data Processor

Our goal is to build a classifier that can detect real networks, but in real-world environments, the ratio of malicious and benign domains is extremely imbalanced. Therefore, we build a module in our system to oversample the imbalanced samples in the training sets to generate balanced samples.

SMOTE is one of the classic methods for solving imbalanced datasets, it is an improved method based on random oversampling. SMOTE constructs new sample points by nearest neighbors of sample points, adding sample points information, which can effectively prevent data over-fitting. It performs the following three steps: Firstly, it chooses a random sample  $\vec{a}$  from minority class. Secondly, choose  $\vec{b}$  from its nearest minority class neighbors. Finally, randomly interpolate  $\vec{x}$  and  $\vec{b}$  to generate the new sample  $\vec{x}$  through this formula  $\vec{x} = \vec{a} + \vec{w} \times (\vec{b} - \vec{a})$ , where  $\vec{w}$  is a random weight in  $[0,1]$ . But SMOTE cannot overcome the problem of data distribution, it can easily cause samples distribution marginalization problem and generates noise.

We propose an improved SMOTE method called KMSMOTE that combines K-means clustering algorithm and SMOTE oversampling algorithm. It counters small separation problems by increasing the number of sparse minority class samples, which can avoid noise by oversampling only in the "safe" region. KMSMOTE method consists of three steps.

1. In the clustering step, use K-means clusters the input samples into  $k$  groups.
2. In the filtering step, calculate the *imbalanceRatio* by  $imbalanceRatio(f) \leftarrow \frac{majorityCount(c)+1}{minorityCount(c)+1}$  of each cluster, if the value of *imbalanceRatio* of cluster less than the imbalance ratio threshold, then put it into *filteredClusters*. By this step, we get *filteredClusters* containing a large number of minority class samples are reserved. Select each cluster  $f$  from *filteredClusters* to calculate *densityFactor* with formula  $densityFactor(f) \leftarrow \frac{minority\ count(f)}{average\ minority\ distance(f)^{de}}$ . Then we get the *sparsityFactor* by  $sparsityFactor(f) \leftarrow \frac{1}{densityFactor(f)}$ . We can get the *samplingWeight* by  $samplingWeight(f) \leftarrow \frac{sparsityFactor(f)}{sparsitySum}$  which *sparsitySum* is the sum of all filtered clusters’ *sparsityFactor*.



3. In the oversampling step, apply SMOTE to the selected clusters for oversampling by applying formula  $SMOTE(f, numberOfSamples, knn)$  where  $numberOfSamples$  is computed by  $samplingWeight$ .

In this algorithm, it is important to find the appropriate  $k$  value to ensure the validity of KMSMOTE. According to our experimental datasets, we set  $k$  to 10 in the experiment.

### 3.4 Domains Classifier

Our classifier is based on CatBoost algorithm [24]. CatBoost algorithm is an implementation of Gradient Boosting. The two main features in CatBoost algorithm are: it derives ordered boosting, which is an improvement to the standard Gradient Boosting algorithm, which can avoid target leakage; it is a novel algorithm to deal with categorical features. These features are designed to solve prediction shifts, which are common in gradient boosting algorithms. Classification and regression trees cannot handle discrete features with categorical values, but CatBoost can support categorical features. An effective way to deal with categorical features  $i$  is to replace the category  $x_k^i$  of  $k$ -th training example with some Target Statistic (TS)  $\hat{x}_k^i$ . This value TS is calculated as follows:  $\hat{x}_k^i = \mathbb{E}(y \mid x^i = x_k^i)$ . CatBoost smoothes low-frequency categories through some prior  $p$ , and the estimates of these low-frequency categories are noisy. It first randomly sorts all samples and then converts the values of the categorical features into numerical values using the formula

$$x_k^i \hat{x}_k^i = \frac{\sum_{j=1}^{p-1} [x_{\sigma_j, k} = x_{\sigma_p, k}] Y_{\sigma_j} + a \cdot P}{\sum_{j=1}^{p-1} [x_{\sigma_j, k} = x_{\sigma_p, k}] + a} \quad (1)$$

where  $p$  is the priori term, which is usually be setted as the average target value in the dataset [25] and  $a$  is usually a weighting factor greater than zero. In order to solve the prediction shift that occurs in each step of gradient boosting, CatBoost proposes ordered Boosting. The method is as follows: first randomly generate a 1- $n$  permutation named  $\sigma$  and maintain  $n$  different supporting models  $M_1, \dots, M_n$ ,  $M_i$  is the training model obtained using only the first  $i$  samples in the permutation. At each step of the iteration, in order to obtain the estimated value of the  $j$  sample residual, use the model  $M_{j-1}$  to estimate.

## 4 Experiments

### 4.1 Data

**Active DNS traffic data.** We build a domain list by ground truth which includes the benign domain lists and malicious domain lists. The lists of benign domains we select are the Second Level Domains of popular domains, such as the Alexa Top Sites [21] and the Cisco Umbrella list, and domains must survive on these lists for more than one year.

Malicious domains lists are extracted from public blacklists, such as Malware-domainlist, etc. These malicious domain lists contain various malicious activities, and we choose Second Level Domains from multiple blacklists to construct our blacklists. And through McAfee Site Advisor, and Google Safe Browsing and other reputation systems for secondary confirmation to ensure the accuracy of the domains label of our list. We obtain active DNS traffic data by sending domain requests to domains of our domain list.

**DNS related external data.** We collect DNS related external data from corresponding databases, such as obtain domain name registration records by the Whois protocol; query rich resource record database to obtain rich related resource records, such as MX, NS, PTR, etc; query the maliciousness of IPs from public IP blacklists.

We build an initial domain list contains 10000 benign domains and 6000 malicious domains and obtain an imbalanced DNS dataset by collecting Active DNS traffic data and external DNS-related data.

## 4.2 Malicious Domains Detection Performance

We randomly select a dataset consisting of DNS data of 10,000 benign domains and 1000 malicious domains from our imbalanced dataset, the proportion of imbalance of this dataset is large, which is 10. We then perform 10-fold cross-validation on this experimental dataset. The training data is initially divided into 10-fold. In each experiment, 9-fold data is used for training and 1-fold data is used for testing. We calculate the average of the ten results of experiments and showed them in Table 2.

As we can see in Table 2, KSDom obtains good performance, the average accuracy is 0.9842, the average F1-Score is 0.9838, and the average precision obtains 0.9926. We then use the trained classifier to classify unlabeled domains and add labeled domains to our domain list to better train the model. We extract rich features to build our classifier. These features represent the domains from different perspectives and have different effects on the generation of a classifier. We test the detection performance of each feature by selecting 70% samples for training and 30% samples for testing from our initial dataset and repeat the procedure for 10 times. Catboost provides a feature importance algorithm, and we can use `get_feature_importance()` method to get the importance of the features we selected, the results are shown in Table 1. From Table 1, we can see that features such as the number of malicious IP addresses, the distribution of consonant characters and the number of labels of the website's contents are all of great importance.

## 4.3 Performance Comparison with Different Sparsity

To test the effectiveness of KSDom, in this section, we compare KSDom with three classic classification methods based on the same dataset: Support Vector Machine (SVM), Gradient Boosting Decision Tree (GBDT) and eXtreme Gradient Boosting (XGBoost). Before using these three classification methods for

training, first perform one-hot encoding on categorical features, CatBoost does not need to encode them because it can process categorical features.

Table 2: Detection results with different methods

| Metrics             | SVM      | GBDT    | XGBoost | KSDom          |
|---------------------|----------|---------|---------|----------------|
| <b>accuracy</b>     | 0.9484   | 0.9478  | 0.9481  | <b>0.9842</b>  |
| <b>precision</b>    | 0.9508   | 0.9507  | 0.9507  | <b>0.9926</b>  |
| <b>F1-Score</b>     | 0.9641   | 0.9637  | 0.9639  | <b>0.9838</b>  |
| <b>running time</b> | 174.564s | 94.912s | 89.074s | <b>52.518s</b> |

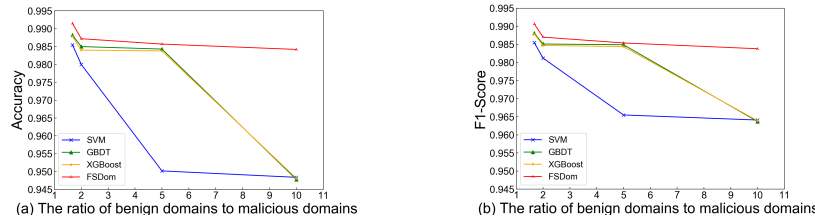


Fig. 2: Accuracy and F1-Score in different ratio of benign domains to malicious domains

We perform 10-fold cross-validation to the 4 experiments to compare the performance of KSDom with other classifiers in different ratios of malicious domains and benign domains. In these 4 experiments, we set the number of benign domains in the training set is 10000, and the number of malicious domains is 6000, 4000, 2000 and 1000 (the imbalanced ratio of benign domains to malicious domains is 1.67, 2.5, 5, 10). Figure 2 shows the average of 10-fold cross-validation results.

From Figure 2, we can see that as the imbalance ratio increases, the accuracy and F1-Score of other classifiers are extremely reduced, but KSDom still maintains good performance.

We can get further insight into the comparison of KSDom and other classification methods from Table 2. Table 2 shows the accuracy, precision, F1-Score and the running time of each method. It can be seen that the accuracy, precision and F1-Score of KSDom are higher than other classification methods, and the running time is shorter than others. For example, compared to SVM, GBDT, XGBoost, the accuracy of KSDom is increased by approximately 3.58%, 3.64% and 3.61%. As we can see, KSDom still maintains relatively good performance when the imbalanced ratio increases. It yields accuracy: 0.9915, F1-Score: 0.9907 when the imbalanced ratio is 1.67 and still obtains accuracy: 0.9842, F1-Score: 0.9838 when the imbalanced ratio increases to 10. As for the other classification methods, they can get excellent performance with a small imbalanced ratio, yet the accuracy and F1-score both decrease significantly as the imbalanced ratio increases. The reason behind this is that KSDom has the KMSMOTE algorithm, which can effectively handle imbalanced data sets and make its trained models

have higher performance. The classifier used by KSDOM is Catboost, which has excellent performance. Catboost has an ordered boosting algorithm and a module that can process classified features, which improve its performance. CatBoost algorithm also implements a symmetric tree that can deal with the correlation between features, which helps avoid data overfitting and greatly accelerates predictions.

#### 4.4 Impact of Oversampling Algorithm

In the solution of imbalanced datasets, the two classic methods in the resampling method are EasyEnsemble in undersampling and SMOTE in oversampling. Since EasyEnsemble combines undersampling and ensemble learning, we use SMOTE and CatBoost algorithm combination to compare with our KSDom.

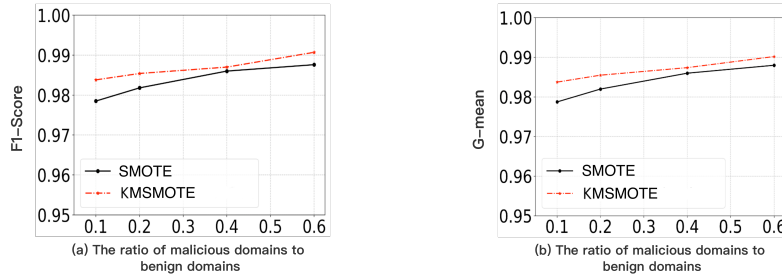


Fig. 3: F1-Score and G-mean in different ratio of malicious domains to benign domains

To prove that the K-means of our SMOTE method is better than SMOTE, we compare the two methods with the following datasets: we establish five datasets, each of which contains 10000 benign domains and 1000, 2000, 4000, 6000 malicious domains (the imbalance ratio between malicious domain and benign domain is 0.1, 0.2, 0.4 and 0.6). The experimental results are shown in Figure 3.

Traditional evaluation metrics used to evaluate classifier performance are not applicable to imbalanced samples, such as accuracy, precision, etc. There are a number of indicators that are specifically used to assess imbalanced data, such as the F1-Score, G-mean. In this experiment, we use F1-Score to measure the classification performance of the minority class, while G-means are used to measure the overall classification performance of the datasets.

As shown in Figure 3, our KMSMOTE method performs better than SMOTE in different ratios datasets. Especially when dealing with extremely imbalanced datasets, such as when the ratio is 0.1, the F1-Score and G-mean of KMSMOTE are both higher than SMOTE. Therefore, we can conclude that our KMSOMTE method is superior to SMOTE in dealing with imbalanced data.

## 5 Conclusion

In this paper, we propose a system for detecting malicious domains named KSDom, which can effectively detect the malicious domains from imbalanced DNS traffic. KSDom collects DNS traffic from the data collector and employs a KMSMOTE method to process the imbalanced DNS traffic, then uses our classifier based on CatBoost algorithm to detect malicious domains. The comparative experimental results show that KSDom has higher accuracy than other classifiers. It performs well even when the datasets imbalance ratio reaches 10:1. In our extensive evaluation, we verified KSDom's good performance and robustness.

## Acknowledgment

This research is supported by National Key Research and Development Program of China (No.2019QY1303, No.2019QY1302, No.2018YFB0803602), and the Strategic Priority Research Program of the Chinese Academy of Sciences (No.XDC02040100), and National Natural Science Foundation of China (No.61802404). This work is also supported by the Program of Key Laboratory of Network Assessment Technology, the Chinese Academy of Sciences; Program of Beijing Key Laboratory of Network Security and Protection Technology.

## References

1. Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster. Building a dynamic reputation system for dns. In USENIX security symposium, pages 273–290, 2010.
2. Leyla Bilge, Sevil Sen, Davide Balzarotti, Engin Kirda, and Christopher Kruegel. Exposure: A passive dns analysis service to detect and report malicious domains. *ACM Transactions on Information and System Security (TISSEC)*, 16(4):14, 2014.
3. Issa M Khalil, Bei Guan, Mohamed Nabeel, and Ting Yu. A domain is only as good as its buddies: Detecting stealthy malicious domains via graph inference. In *Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy*, pages 330–341. ACM, 2018.
4. Issa Khalil, Ting Yu, and Bei Guan. Discovering malicious domains through passive dns data graph analysis. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 663– 674. ACM, 2016.
5. Xiaoqing Sun, Mingkai Tong, Jiahai Yang, Xinran Liu, Heng Liu. HinDom: A Robust Malicious Domain Detection System based on Heterogeneous Information Network with Transductive Classification. *RAID 2019*: 399-412
6. 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 *USENIX security symposium*, volume 12, 2012.
7. Schiavoni, S., Maggi, F., Cavallaro, L., Zanero, S.: Phoenix: DGA-based botnet tracking and intelligence. In: Dietrich, S. (ed.) *DIMVA 2014*. LNCS, vol. 8550, pp. 192–211. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-08509-8\\_11](https://doi.org/10.1007/978-3-319-08509-8_11)

8. Shuang Hao, Alex Kantchelian, Brad Miller, Vern Paxson, and Nick Feamster. 2016. PREDATOR: Proactive recognition and elimination of domain abuse at time-of-registration. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security. 1568–1579.
9. C. Zhou, K. Chen, X. Gong, P. Chen, and H. Ma, “Detection of fast-flux domains based on passive DNS analysis,” *Acta Scientiarum Naturalium Universitatis Pekinensis*, vol. 52, no. 3, pp. 396–402, 2016.
10. Babak Rahbarinia, Roberto Perdisci, and Manos Antonakakis. Segugio: Efficient behavior-based tracking of malware-control domains in large isp networks. In Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on, pages 403–414. IEEE, 2015.
11. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.
12. S. Maldonado, R. Weber, and F. Famili, “Feature selection for high-dimensional class-imbalanced data sets using support vector machines,” *Information Sciences*, vol. 286, pp. 228–246, 2014.
13. N. Nikolaou, *Cost-Sensitive Boosting: A Unified Approach*, The University of Manchester, 2016.
14. X.-Y. Liu, J. Wu, and Z.-H. Zhou, “Exploratory undersampling for class-imbalance learning,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 2, pp. 539–550, 2009.
15. Liu, and X. Ying . ”An Empirical Study of Boosting Methods on Severely Imbalanced Data.” *Applied Mechanics and Materials* 513-517(2014):2510-2513.
16. Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.
17. Zhang W , Wang J . A Hybrid Learning Framework for Imbalanced Stream Classification[C]// IEEE International Congress on Big Data. IEEE, 2017.
18. Wang Q . A Hybrid Sampling SVM Approach to Imbalanced Data Classification[J]. *Abstract and Applied Analysis*, 2014, 2014(5):1-7.
19. Zhenyan L , Yifei Z , Pengfei Z , et al. An Imbalanced Malicious Domains Detection Method Based on Passive DNS Traffic Analysis[J]. *Security and Communication Networks*, 2018, 2018:1-7.
20. Yury Zhauniarovich, Issa Khalil, Ting Yu, and Marc Dacier. 2018. A Survey on Malicious Domains Detection through DNS Data Analysis. *ACM Comput. Surv.* 51, 4, Article 67 (July 2018), 36 pages.
21. Amazon Web Services, Inc. AWS | Alexa Top Sites - Up- to-date lists of the top sites on the web. <https://aws.amazon.com/alexa-top-sites/>, 2019. [Online].
22. Athanasios Kountouras, Panagiotis Kintis, Charles Lever, Yizheng Chen, Yacin Nadji, David Dagon, and Manos Antonakakis. 2016. Enabling network security through active DNS datasets. In Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses. 188–208.
23. Iria Prieto, Eduardo Magaña, Daniel Morató, and Mikel Izal. 2011. Botnet detection based on DNS records and active probing. In Proceedings of the International Conference on Security and Cryptography. 307–316.
24. Prokhorenkova L , Gusev G , Vorobev A , et al. CatBoost: unbiased boosting with categorical features[J]. 2017.
25. D. Micci-Barreca. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter*, 3(1):27–32, 2001.