Hindawi
Security and Communication Networks
Volume 2018, Article ID 6510381, 7 pages
https://doi.org/10.1155/2018/6510381

Research Article

An Imbalanced Malicious Domains Detection Method Based on Passive DNS Traffic Analysis

Zhenyan Liu, Yifei Zeng, Pengfei Zhang, Jingfeng Xue, Ji Zhang, and Jiangtao Liu

Beijing Key Laboratory of Software Security Engineering Technology, School of Software, Beijing Institute of Technology, Beijing 100081, China

Correspondence should be addressed to Zhenyan Liu; zhenyanliu@bit.edu.cn

Received 9 January 2018; Accepted 19 April 2018; Published 20 June 2018

Academic Editor: Yuqing Zhang

Although existing malicious domains detection techniques have shown great success in many real-world applications, the problem of learning from imbalanced data is rarely concerned with this day. But the actual DNS traffic is inherently imbalanced; thus how to build malicious domains detection model oriented to imbalanced data is a very important issue worthy of study. This paper proposes a novel imbalanced malicious domains detection method based on passive DNS traffic analysis, which can effectively deal with not only the between-class imbalance problem but also the within-class imbalance problem. The experiments show that this proposed method has favorable performance compared to the existing algorithms.

1. Introduction

With the rapid development of the Internet and information technology, network security threats are escalating, the security of cyberspace is becoming more and more complex and hidden, the risk of network security is increasing, and various network malicious attacks emerge endlessly. In these network malicious attacks, most of them are based on DNS (Domain Name System). The reason why DNS can provide an available infrastructure for attackers is that it is open and ease of use.

The core of the network malicious attack based on DNS is C&C (Command and Control) server. By means of the C&C server, the attackers can order remote hosts to perform malicious activities, such as spamming, phishing, DDOS (Distributed Denial of Service), and distributing malware which may be used to steal information, disrupt computer, extort money, etc. Therefore, it is urgent to detect this kind of malicious domain of C&C server and further take corresponding countermeasure.

It is very popular to employ the classification algorithm in machine learning to detect malicious domains in the current research [1, 2]. However, these existing studies pay no or little attention to the problem of imbalanced data. In fact, the actual DNS traffic is inherently imbalanced, in which most of the cases are benign and far fewer cases are malicious. As a result, this tends to construct an imbalanced training dataset in which there are many more samples of some categories than others. When learning from an imbalanced dataset, class information must be considered; otherwise the classifier will be overwhelmed by the majority classes and ignores the minority ones, and then the overall classification performance will undoubtedly be degraded. To address this shortfall, this paper will propose an imbalanced malicious domains detection method which can build malicious domains detection model by learning imbalanced dataset based on passive DNS traffic analysis.

In this paper we make the following contributions:

(1) We especially focus on learning from the imbalanced data in the malicious domains detection field. And the latest research progress of learning from the imbalanced data in other fields is invited in the malicious domains detection field.

(2) We construct the stronger discriminative features to profile malicious domains based on passive DNS traffic analysis.

(3) We propose an improved imbalanced malicious domains detection method which is an extension of EasyEnsemble and demonstrate its favorable performance by the comparative experiments.

The remainder of this paper is organized as follows. In Section 2, we briefly review related work. Section 3 describes how to profile malicious domains based on passive DNS traffic analysis. We elaborate on an imbalanced malicious domains detection method in Section 4. Section 5 presents our comparative experiments of this new method. Finally, we conclude the paper in Section 6.

2. Related Work

2.1. Learning from Imbalanced Data. Although rarely in network security, learning from the imbalanced data has already made considerable progress in other fields. In general, there are three ways to tackle the imbalanced learning problem. The first one is from the data perspective, which mainly uses resampling approaches to modify the class distribution of the data. The second one is from the algorithm perspective, which mostly focuses on optimizing various algorithms, such as SVM (Support Vector Machine), Decision Tree and Neural Network, based on cost-sensitive learning which considers the costs associated with misclassifying samples [3]. In addition, some researches also utilize one-class learning [4] which is particularly useful when used on extremely imbalanced data sets. The third one is from data feature perspective, which can build a fair feature space attaching much weight to the minority classes by means of some improved feature selection methods. This third approach is applied in many applications, including fraud/churn detection, text categorization, medical diagnosis, detection of software defects, and many others [5].

Most researches have been focused on the first approach, resampling which is more practical than the other two approaches. The resampling includes undersampling, oversampling, and the integration of undersampling and oversampling [6]. The key idea of undersampling is to remove the majority class samples from the original data set, and the key idea of oversampling is to append the minority class samples to the original data set.

The simplest resampling technique is random. But random undersampling can potentially rmove certain important samples, and random oversampling can lead to overfitting. Various improved undersampling algorithms, including EasyEnsemble and BalanceCascade, have been proposed [7]. Both methods utilize ensemble learning to overcome the deficiency of information loss introduced in the traditional random undersampling, since ensemble learning is based on multiple subsets which contain more information than a single one [8]. The famous improved oversampling algorithms are SMOTE (Synthetic Minority Oversampling Technique) [9] and its variants, such as Borderline-SMOTE [10] and ADASYN (Adaptive Synthetic Sampling) [11]. They devote to create the excellent artificial minority class samples using different strategies.

In practical application, when the samples of the minority classes are absolutely rare, oversampling is generally employed to increase the samples of the minority classes. Or else when the samples of the minority classes are relatively rare, undersampling is generally employed to decrease the samples of the majority classes.

2.2. Malicious Domain Detection Based on Passive DNS Traffic Analysis. The majority of detection methods based on DNS traffic are data-driven, most commonly having machine learning algorithms at their core. These methods require accurate ground truth of both malicious and benign DNS traffic for model training as well as for the performance evaluation [12]. The methods of DNS data collection can be generally divided into two subcategories: active and passive. *Active* method obtains DNS data by deliberately sending DNS queries and record the corresponding DNS responses, while *passive* method is passively to backup real DNS queries and responses.

Compare with *active* DNS data collection, *passive* DNS data collection is more representative and more comprehensive. As a result, the detection of malicious domain based on passive DNS traffic analysis has received increasing attention from the research community over the past decade. “Passive DNS” was invented by Weimer [13] in 2004. After that, many researchers have an insight into the important value of passive DNS when doing incident response investigations. And many passive DNS systems have developed, in which the most famous and popular one is DNSDB from Farsight Security. Farsight collects passive DNS data from its global sensor array, and then filters and verifies the DNS transactions before inserting them into the DNSDB [14]. The trends within this set are believed to be representative of Internet-wide trends and therefore provide valuable insight.

Antonakakis et al. [1] proposed a dynamic reputation system for DNS, called Notos, to automatically assign a low reputation score to a malicious domain. To measure a number of statistical features of a domain, Notos used historical DNS information collected passively from multiple recursive DNS resolvers distributed across the Internet. Bilge et al. [2] introduced a passive DNS analysis approach and a detection system, EXPOSURE, to detect domain names that are involved in malicious activities. The data that EXPOSURE used for the initial training consist of DNS traffic from the real-time response data from authoritative Name Servers located in North America and in Europe.

Perdisci et al. [15] presented FluxBuster, a novel detection system that used a purely passive approach for detecting and tracking malicious flux networks. FluxBuster is based on large-scale passive analysis of DNS traffic generated by hundreds of local recursive DNS (RDNS) servers located in different networks and scattered across several different geographical locations. Zhou et al. [16] proposed a model which can detect Fast-Flux Domains using random forest algorithm. It used passive DNS to log domain name query history of real campus network environment.

Analyzing these existing related works, we discovered that most of them are to collect DNS traffic in a period time to form a passive DNS set. This kind of passive DNS set is only a DNS data fragment and needs more collection cost. While DNSDB is relatively comprehensive, as a result, we determined to use the passive DNS traffic from DNSDB in this paper.

3. Profiling Malicious Domains Based on Passive DNS Traffic Analysis

To profile malicious domains, based on passive DNS traffic analysis we extract two groups features of malicious domains: static lexical features and dynamic DNS resolving features. Static lexical features mainly origin from the lexical information of domain name. Dynamic DNS resolving features are constructed based on DNS response attributes. Table 1 gives an overview of these features.

The results of statistical analysis of some features are selected to show in Figure 1. From these, we can find that these features have the stronger ability to distinguish the malicious domains from the benign ones.

In this section, we will present 12 static lexical features and 4 dynamic DNS resolving features and the motivation that we construct these features to profile malicious domains.

3.1. Static Lexical Features. To avoid detection, the attackers generally employ domain generation algorithms (DGA) to dynamically produce a large number of random domain names. The lexical features of these malicious domain names are largely different from benign domain names. We construct 12 static lexical features to profile malicious domains.

So far the short domain names have been almost registered; therefore the majority of malicious domain names generated by DGA are longer than benign domain names. And max length of labels (i.e., parts delimited by dots) in subdomain of malicious domain names is also commonly longer. So we construct two features based on the length measure: first, length of domain name (Feature 1), and second, max length of labels in subdomain (Feature 2).

The most distinctive property of domain names generated by DGA is that the distribution of characters is random. We know that information entropy is defined as the average amount of information produced by a stochastic source of data [17]. So, we employ information entropy to measure the disorder of characters.

Let d be a domain name and m be the number of distinct characters in d. We define entropy (d) as character entropy of d (Feature 3).

$$Entropy(d) = -\sum_{\mathrm{i}=1}^{\mathrm{m}} \left(\frac{\mathrm{count}(\mathrm{a_i})}{\mathrm{length}(d)}\right) \log_2 \left(\frac{\mathrm{count}(\mathrm{a_i})}{\mathrm{length}(d)}\right) \quad (1)$$

where $\mathrm{a_i}$ $(\mathrm{i} = 1 \ldots \mathrm{m})$ means a character in d, count($\mathrm{a_i}$) is the number of $\mathrm{a_i}$ in d, and length(d) is the length of d,.

If the character entropy value of d is greater, then more likely d will be identified to be malicious.

In addition, malicious domain names are used by malwares not by human, so they are not easy-to-remember or human pronounceable. Thus the appearance of numerical and alphabetic characters in malicious domain names is also very important indicative signs. With this insight, we construct five features as follows: number of numerical characters (Feature 4), ratio of numerical characters (Feature 5), conversion frequency of numerical and alphabetic character (Feature 6), max length of continuous numerical characters (Feature 7), max length of continuous alphabetic characters (Feature 8), and max length of continuous same alphabetic characters (Feature 9).

As we all know, the consonant letters in the English alphabet are much more than the vowel letters. Therefore, in random malicious domain names, the ratio of vowels (Feature 10) is smaller, the length of continuous consonants (Feature 11) is longer, and conversion frequency of vowel and consonant (Feature 12) is very higher.

3.2. Dynamic DNS Resolving Features. The Internet-scale attacks using DNS leave unavoidably a trail of footmarks which are hidden into the DNS resolving records, so we may mine these footmarks (i.e., DNS resolving features) to profile malicious domains. In this section, we will present 4 dynamic resolving features origin from the DNS resolving records.

In order to evade blacklists and resist takedowns, the DNS answer that is returned by the server for a malicious domain generally consists of multiple DNS A records (i.e., Address records) or NS records (i.e., Name Server records). And the slippery attackers do not usually target specific Name Server or IP ranges. Therefore, we construct four statistical features as follows: number of distinct A records (Feature 13), IP entropy of domain name (Feature 14), number of distinct NS records (Feature 15), and similarity of NS domain name (Feature 16).

Number of distinct A records (Feature 13) records the total number of IP addresses resolved in DNSDB. Furthermore, IP entropy of domain name (Feature 14) is constructed to measure the dispersion of these IP addresses resolved. Let d be a domain name, S be the set of these IP addresses resolved, and n be the number of distinct IP/16 prefixes in S. We define $IP\_Entropy(d)$ as IP entropy of domain name (Feature 14).

$$IP\_Entropy(d) = -\sum_{\mathrm{i}=1}^{\mathrm{n}} \left(\frac{\mathrm{count}(\mathrm{ipx_i})}{|\mathrm{S}|}\right) \log_2 \left(\frac{\mathrm{count}(\mathrm{ipx_i})}{|\mathrm{S}|}\right) \quad (2)$$

where $\mathrm{ipx_i}$ $(\mathrm{i} = 1 \ldots \mathrm{n})$ means an IP/16 prefix in S, count($\mathrm{ipx_i}$) is the number of $\mathrm{ipx_i}$ in S, and |S| is the size of S.

If the IP entropy value of d is greater, then more likely d will be identified to be malicious.

Number of distinct NS records (Feature 15) records the total number of Name Servers resolved in DNSDB. Furthermore, Similarity of NS domain name (Feature 16) is constructed to measure the difference of these Name Servers resolved. We calculate the Edit Distance between every pair of Name Server names of a domain, and then the average of these distances is defined as the similarity of NS domain name. If the similarity of NS domain name of d is bigger, then more likely d will be identified to be malicious.

4. An Imbalanced Malicious Domains Detection Method

Almost all classification algorithms seem to be powerless to learn from an extremely imbalanced training data set. In consideration of the actual imbalanced distribution of

Table 1: An overview of domain features.

Feature group	No.	Feature Name	Malicious domain profile
Static lexical features	1	Length of domain name	Longer
	2	Max length of labels in subdomain	Longer
	3	Character entropy	Greater
	4	Number of numerical characters	Higher
	5	Ratio of numerical characters	Higher
	6	Conversion frequency of numerical and alphabetic character	Higher
	7	Max length of continuous numerical characters	Shorter
	8	Max length of continuous alphabetic characters	Longer
	9	Max length of continuous same alphabetic characters	Shorter
	10	Ratio of vowels	Lower
	11	Max length of continuous consonants	Longer
	12	Conversion frequency of vowel and consonant	Higher
Dynamic DNS resolving features	13	Number of distinct A records	Higher
	14	IP entropy of domain name	Higher
	15	Number of distinct NS records	Higher
	16	Similarity of NS domain name	Bigger

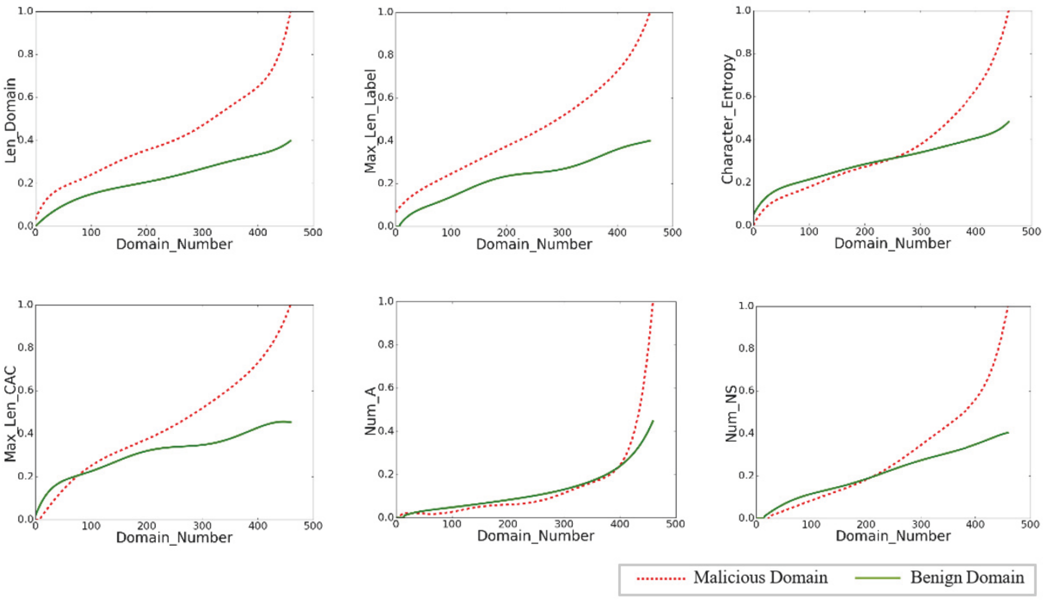

Figure 1: The results of statistical analysis of some features.

DNS traffic data (i.e., malicious domains are relatively rare), inspired by existing methods, our research focuses on the combination of undersampling and ensemble learning.

In existing methods, EasyEnsemble [7] is a typical improved algorithm combining undersampling with ensemble learning. As we know, the main deficiency of undersampling is that potentially useful information contained in the unselected examples is neglected. To remedy this deficiency, EasyEnsemble incorporates ensemble learning into undersampling.

The idea behind EasyEnsemble is quite simple. Given the majority class instances set N_{maj} and the minority class instances set N_{min}, this method independently samples several subsets $N_1, N_2, \ldots\ldots, N_T$ from N_{maj}, where $|N_i| = |N_{\text{min}}|$ $(i = 1, 2, \ldots\ldots, T)$. For each subset N_i, a base classifier is trained using N_i and N_{min}. All base classifiers are combined for the final decision. Remarkably, many learning algorithms can be employed to generate the base classifier.

EasyEnsemble make better use of the majority class than undersampling by ensemble learning, so it is very helpful for between-class imbalance learning. However, EasyEnsemble ignores within-class imbalance, especially for the majority class. That is, in the majority class some instances are highly similar which may form several clusters, and more other

```
(1) {Input: A set of minority class examples $N_{\text{min}}$, a set of majority class examples $N_{\text{maj}}$,
    $|N_{\text{min}}| < |N_{\text{maj}}|$, the number of subsets $T$ to sample from $N_{\text{maj}}$}
(2) $N_{\text{maj}}$ are clustered into several small groups $G_1, G_2, \ldots.$ by HAC
(3) $i \leftarrow 0$
(4) **repeat**
(5) $i \leftarrow i + 1$
(6) Select randomly $(\alpha_j|G_j|)$ instances from each cluster $G_j$ $(j = 1, 2, \ldots)$ with a total of K
(7) Select randomly $|N_i|$ -K instances from $N_{\text{maj}}$- $\Sigma G_j$
(8) Combine the dataset sampled from step (6) and (7) to form a subset $N_i$, where $|N_{i|} = |N_{\text{min}}|$
(9) Learn $H_i$ using $N_i$ and $N_{\text{min}}$, $Hi$ is a base classifier employed Decision Tree
(10) **until** $i = T$
(11) Output: An ensemble $H(\boldsymbol{x}) = \text{argmax}_c \sum_{i=1}^{\text{T}} I(Hi(\boldsymbol{x}) = c)$
```

Algorithm 1: The HAC_EasyEnsemble algorithm.

instances are almost unique. This kind of phenomena is commonly called "long-tailed distribution" in the statistical sense.

We should select a representative subset from each cluster and combine them with a subset selected randomly from the other unique instances set to form a preliminary subset. According to this idea, we proposed an improved EasyEnsemble method to learn imbalanced DNS traffic data.

In this novel method, firstly the instances in the majority class are clustered together in several small groups $G_1, G_2, \ldots\ldots$ by Hierarchical Agglomerative Clustering (HAC). For each cluster G_j $(j = 1, 2, \ldots\ldots)$, according to the size of G_j, we select randomly several instances with a total of K. And then we select randomly $|N_i|$ -K $(i = 1, 2, \ldots\ldots, T)$ instances from N_{maj}- ΣG_j to form a subset N_i, where $|N_i| = |N_{\text{min}}|$. Base classifier H_i is trained using N_i and N_{min}. All T base classifiers are combined for the final decision. Note that Decision Tree algorithm is employed to generate the base classifier.

The pseudocode of the improved EasyEnsemble named HAC_EasyEnsemble is shown in Algorithm 1.

Noted that here I is an indicative function, and c is the class label, if the parameter of I is true, then return 1, or else return 0. In HAC, we may employ various cluster proximity measures which are typically complete link, group average, Ward's method [18], etc. For the complete link, the proximity of two clusters is defined as the maximum of distance (minimum of the similarity) between any two points in the two different clusters. For the group average, the proximity of two clusters is defined as the average pairwise proximity among all pairs of points in the different clusters. For Ward's method, the proximity of two clusters is defined as the increase in the squared error that results when two clusters are merged [19].

5. Experiment

In order to verify the novel HAC_EasyEnsemble algorithm used to learn imbalanced DNS traffic data, we do a series of experiments to compare the performance of HAC_EasyEnsemble and EasyEnsemble based on the same dataset. And we use three different cluster proximity measures in HAC: complete link, group average, and Ward's method.

Originally, we construct an imbalanced training set which contains 6400 benign domains (from alexa.com) and 3000 malicious domains (from cybercrime-tracker.net, malwaredomains.com, and hosts-file.net etc.). The reason for this ratio of malicious domains is that the HAC_EasyEnsemble algorithm is more effective for relatively rare malicious domains, not absolutely. The DNS resolving records of these domains are obtained by DNSDB API, and then 12 static lexical features and 4 dynamic DNS resolving features listed in Section 3 are constructed based on these records.

Commonly the evaluation measures for the imbalanced classification are macroaveraged precision, macroaveraged recall, macroaveraged F1 [20]. Since macroaveraged scores are averaged values over the number of categories, then the performance of classifier is not dominated by major categories. Let P be the precision, R be recall, and m denote the total number of categories, then macroaveraged precision is $(1/m) \sum_{i=1}^{m} \text{P}_i$, macroaveraged recall is $(1/m) \sum_{i=1}^{m} \text{R}_i$, maroaveraged F1 is $(1/m) \sum_{i=1}^{m} \text{F1}_i$, where F1 is2PR/(P + R).

In order to get the number of base classifiers T mentioned in Section 4 of HAC_EasyEnsemble classification model, we firstly do a series of experiments. In these experiments we set different T for HAC_EasyEnsemble classification model, then we observe the error rate of classification in different T. Figure 2 shows the relationship between the number of base classifiers of HAC_EasyEnsemble classification model and the error rate of classification.

From Figure 2, we can find that when the number of base classifiers equals approximately 10, the error rate of classification tends to be unchanged. Consequently, in the next comparing experiments, the number of base classifiers is set as 10.

Tenfold cross validation is performed on the experiment dataset. For this purpose, the corpus is initially partitioned into tenfold. In each experiment, ninefold data are used to train while onefold data are used to test. Ten experiment results are showed in Figure 3 and the average value of ten experiment results is reported in Table 2.

Figure 3 gets further insight about the comparison of complete link clustering, group average clustering, Ward's

Table 2: The macroaveraged P, R, and F1 score comparison of four schemes.

	HAC_EasyEnsemble									EasyEnsemble		
	Complete Link			Group Average			Ward's Method			non-clustering		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Benign	0.9712	0.9500	0.9605	0.9774	0.9591	0.9682	0.9837	0.9622	0.9728	0.9534	0.9375	0.9454
Malicious	0.9552	0.9533	0.9542	0.9491	0.9567	0.9529	0.9651	0.9667	0.9659	0.9181	0.9300	0.9240
Macro-ave	0.9632	0.9517	**0.9574**	0.9633	0.9579	**0.9605**	0.9744	0.9645	**0.9694**	0.9358	0.9338	**0.9347**

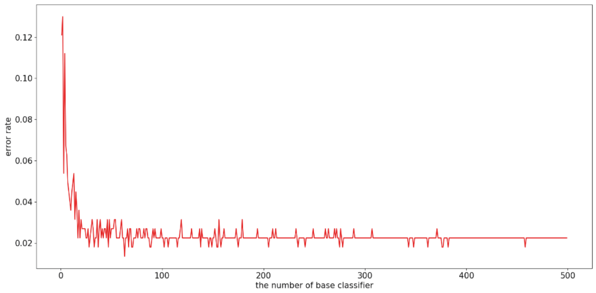

Figure 2: The relationship between the number of base classifiers of HAC_EasyEnsemble and the error rate of classification.

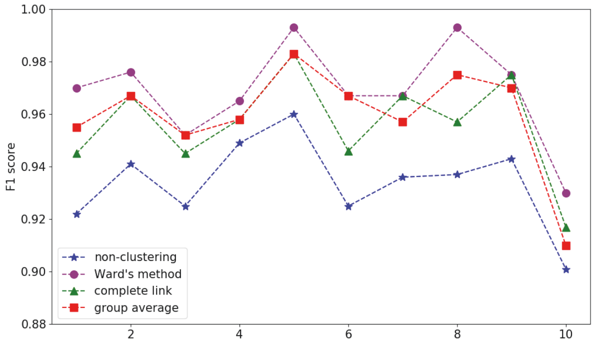

Figure 3: The F1 scores comparison of four schemes.

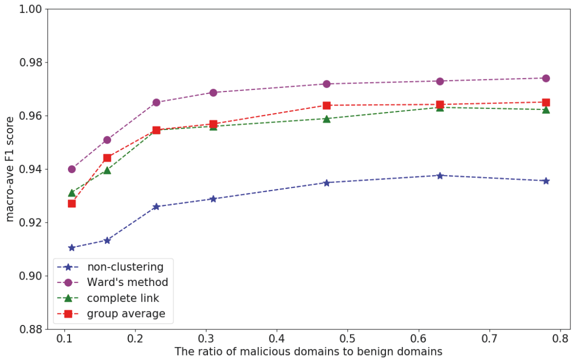

Figure 4: The macroaveraged F1 score in different ratio of malicious domains to benign domains.

method clustering, and nonclustering with line chart form, from which it can be seen that the scores with clustering are nearly higher than ones with nonclustering overall in each experiment. And Ward's method is the best among of them in performance, while complete link and group average are almost in same level.

Table 2 shows the macroaveraged P, R, and F1 score of each scheme. For example, compared to nonclustering, the macroaverage F1 scores of Ward's method clustering, of group average clustering, and of complete link clustering are approximately improved 3.5%, 2.6%, and 2.3%, respectively, and then we can draw a conclusion that sampling with HAC will be very helpful to improve the performance of classifier.

Finally, to find out whether the HAC_EasyEnsemble is able to show its advantage in different ratio of malicious domains, we do the other 6 experiments to compare the detection performance of HAC_EasyEnsemble. In the 6 experiments, the number of benign domains in training set is 6400, and the number of malicious domains is 700, 1000, 1500, 2000, 4000, and 5000, respectively. Figure 4 shows the experimental results.

From Figure 4, we can see that the detection performance of HAC_EasyEnsemble is almost in line with the previous 3000:6400 (see Figure 3 and Table 2) in any other ratio greater than 1000:6400(≈16%). So, if properly used, HAC_EasyEnsemble can be used to detect malicious domains by learning from imbalanced DNS traffic data.

6. Conclusions

In this paper, we proposed an improved version of EasyEnsemble for detecting malicious domains named HAC_EasyEnsemble, which can effectively deal with the within-class imbalance problem in tandem with the between-class imbalance problem, while EasyEnsemble can only deal with the between-class imbalance problem. The key idea of this improvement is to incorporate HAC into undersampling of EasyEnsemble, and three typical cluster proximity measures which are complete link, group average, and Ward's method are also compared by experiments. Moreover, to profile malicious domains, we construct 12 static lexical features and 4 dynamic DNS resolving features based on passive DNS data from DNSDB. The comparative experiments show that the HAC_EasyEnsemble is superior for the malicious domains detection oriented to imbalanced DNS traffic. And it is worth emphasizing that this novel method is extremely suitable for the tasks in which enough malicious domains cannot be obtained in a limited amount of time.

We believe that HAC_EasyEnsemble is an effective method that can help us to cope with cybercrime. As future work, we plan to construct more discriminative features to profile malicious domains and further to enhance the performance of the HAC_EasyEnsemble algorithm.

Data Availability

The authors declare that the data used in our manuscript can be accessed by the following method. Firstly, the benign domain names are downloaded from alexa.com and the malicious domain names are downloaded from cybercrime-tracker.net, malwaredomains.com, hosts-file.net, etc. And then the DNS resolving records of these domains are obtained by DNSDB API with additional charge.

Disclosure

The funders (Dr. Xue and Dr. Shan) were involved in the manuscript editing, approval, or decision to publish.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was financially supported by National Key R&D Program of China (2016YFB0801304) and Scientific Research Project of Beijing Institute of Technology (2017CX02029).

References

[1] M. Antonakakis, R. Perdisci, D. Dagon et al., “Building a dynamic reputation system for DNS,” *Usenix Security Symposium*, pp. 273–290, 2010.

[2] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, “EXPOSURE: a passive DNS analysis service to detect and report malicious domains,” *ACM Transactions on Information and System Security*, vol. 16, no. 4, p. 14, 2014.

[3] N. Nikolaou, *Cost-Sensitive Boosting: A Unified Approach*, The University of Manchester, 2016.

[4] S. Wang and X. Yao, “Relationships between diversity of classification ensembles and single-class performance measures,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 1, pp. 206–219, 2013.

[5] 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.

[6] Y. Peng and J. Yao, “AdaOUBoost: adaptive over-sampling and under-sampling to boost the concept learning in large scale imbalanced data sets,” in *Proceedings of the ACM SIGMM International Conference on Multimedia Information Retrieval (MIR ’10)*, pp. 111–118, March 2010.

[7] 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.

[8] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, “An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics,” *Information Sciences*, vol. 250, no. 11, pp. 113–141, 2013.

[9] 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.

[10] H. Han, W. Y. Wang, and B. H. Mao, “Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning,” in *Proceedings of the International Conference on Intelligent Computing*, vol. 3644 of *Lecture Notes in Computer Science*, pp. 878–887, Springer, 2005.

[11] H. He, Y. Bai, E. A. Garcia, and S. Li, “ADASYN: adaptive synthetic sampling approach for imbalanced learning,” in *Proceedings of the International Joint Conference on Neural Networks (IJCNN ’08)*, pp. 1322–1328, June 2008.

[12] M. Stevanovic, J. M. Pedersen, A. D’Alconzo, S. Ruehrup, and A. Berger, “On the ground truth problem of malicious DNS traffic analysis,” *Computers & Security*, vol. 55, pp. 142–158, 2015.

[13] F. Weimer, “Passive DNS replication,” in *Proceedings of the FIRST Conference on Computer Security Incident*, pp. 1–13, 2004.

[14] https://www.farsightsecurity.com/solutions/dnsdb/.

[15] R. Perdisci, I. Corona, and G. Giacinto, “Early detection of malicious flux networks via large-scale passive DNS traffic analysis,” *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 714–726, 2012.

[16] 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.

[17] https://en.wikipedia.org/wiki/Entropy_(information_theory).

[18] F. Murtagh and P. Contreras, “Algorithms for hierarchical clustering: an overview,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[19] S. Miyamoto, R. Abe, Y. Endo, and J.-I. Takeshita, “Ward method of hierarchical clustering for non-Euclidean similarity measures,” in *Proceedings of the 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR ’15)*, pp. 60–63, IEEE, November 2015.

[20] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*, Cambridge University Press, 2010.