Academic Journal of Computing & Information Science
ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

A Ensemble Learning method for Domain Generation Algorithm Detection

Yihang Zhang

Zhengzhou Foreign Language School, Zhengzhou 450001, Henan, China

ABSTRACT. *A botnet is a one-to-many control network among a master and its infected hosts which are utilized to commit malicious activities like DDoS attacks, mining, posting spam, etc. Attackers used to utilize hard-coded domain names to manipulate the connections between bots and the C&C (command and control) server. However, since this method is easy to ban, attackers currently tend to shuttle in bulk of domains generated by algorithms (DGA domains) to improve their flexibility and bypass the blacklists. To solve this problem, we propose an automated DGA detection system based on machine learning methods. We extract 12 features to represent the differences in character distributions of legal and DGA domains. To improve detection performance and versatility, we also apply ensemble learning methods to the DGA classifier. Experiments on public datasets show that the XGBoost-based classifier surpasses all the other methods in both accuracy and efficiency.*

KEYWORDS: *DGA detection, Machine Learning, Ensemble Learning*

1. Introduction

As one of the most important infrastructure of Internet, Do- main Name System (DNS) is responsible for mapping easy-to- remember domain names to IP addresses. Though supporting most of the current Internet applications, DNS is also abused by malicious attacks like Botnets [1], which has become one of the main threats of the current Internet. With botnets, attackers can launch a series of malicious activities, such as distributed denial of service attack (DDoS), click fraud, sending spam, etc. To avoid being blocked, botnets utilize domain generation algorithms (DGA) to build connections between the C&C server and infected hosts. Currently, DGA has been widely used in various botnets, such as Conficker [2], Mjuyh [3], Torpig [4], etc, making blacklists ineffective. Facing this problem, researchers try to apply machine learning methods to DGA domain detection [5]–[7]. They extract various features like query frequency, the number of related IP addresses, etc, to train a classifier. Aside from basic machine learning models, ensemble learning methods are proposed to build a more comprehensive model. In this work, we focus on the

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

character distribution features of domain names and utilize ensemble learning methods to improve detection performance and versatility.

The main work of this paper is summarized as follows:

- We analyze the character distribution of domain names and extract 12 features to represent the differences be- tween legal domains and DGA domains.
- We implement several DGA classifiers with basic ma- chine learning methods and ensemble methods.
- We evaluate the classifiers with multiple metrics based on public datasets and find that XGBoost-based classifier obtains the best detection performance with the shortest training time.

The remainder of this paper is organized as follows. We introduce the background of DGA domains and the existing work on DGA detection in Section 2. The technical details of feature extraction and detection methods are shown in Section 3. Section 4 presents evaluation metrics and experiment results. We summarize this project and discuss future work in Section 5.

2. Methodology

A. Domain Generation Algorithm

Domain generation algorithm (DGA) is a technical method that uses random characters to generate pseudo-random strings [8]. Attackers can then use these strings as domain names for C&C servers to bypass blacklists and establish connections with remote infected hosts. Specifically, the attacker runs a DGA, randomly registers one or several generated domains and points them to the C&C server. On the victim side, an infected host runs the DGA and queries all the generated domains. Once a registered domain is queried, the host can communicate with the C&C server. The defenders can identify DGA domains through malware sample collection and reverse engineering. However, DGA can generate thousands of do- mains in a short time. It is impossible for defenders to update the blacklists with such a high frequency.

B. Related Work

There are already some researches devoted to DGA domains detection and the methods can be divided into active analysis and passive analysis [5]. Active analysis methods include DNS detection, web content analysis, and manual expert analysis. Passive analysis includes black and white list rule matching, machine learning, and graph theory. Theoretically, the number of DGA domains can be indefinite. However, each additional domain in the blacklist will add a burden to the server. Therefore, applying machine learning methods to DGA domain detection has become an effective way to countermeasure against attackers. Notos [9] and Exposure [7] use network features and zone features to calcuate reputation scores for domains. Kopis [10] utilize traffic among top-level- domain servers to gather more information while Pleiades [6] only focus on NXDomain traffic to improve efficiency. Though

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

these work achieve good performance, the DNS- related data, like traffic, logs and other private information are hard to obtain. FANCI [11] solved this problem by only using domain name strings. It extracts 21 features like the length of domain names, the number of subdomains, etc to separate DGA domains and legal ones. Our project belongs to the same category. Besides, we utilize ensemble learning methods to get a more comprehensive DGA classifier.

3. Project

Figure 1 illustrates the workflow of our project. We first train the classifier with features (e.g. the length of domain name strings, entropy, etc.) extracted from the labeled domain dataset. Then, the classifier can be used to evaluate unknown domains and generate detection reports. As we can see, feature extraction and detection algorithm selection are the two key parts of this project, which we will elaborate on next.

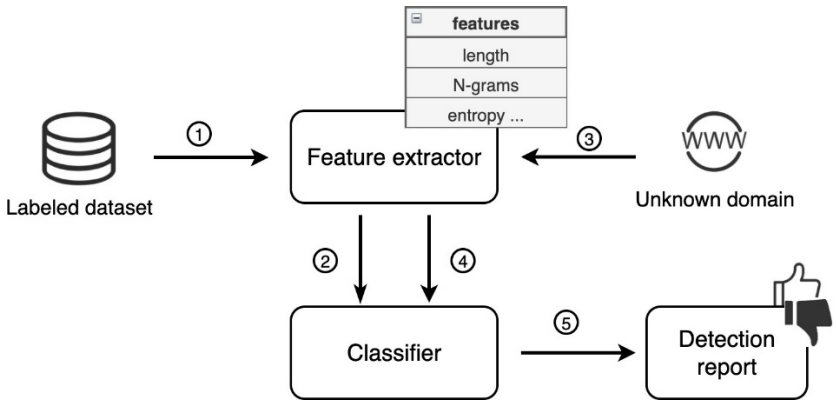

Figure. 1 Workflow of the DGA detection system

3.1. Feature Extraction

When registering a domain name, legal users tend to choose a combination of a few words to fit their business. For example, they may take *facebook.com*, *linkedin.com*, *taobao.com*, etc. These domains usually have specific se- mantics and are easy to remember. On the other hand, DGA domains are generated by algorithms with random seeds, which makes them mostly disordered and unreadable. Therefore, we extract the following 12 features to represent the differences in character distribution and help distinguish DGA domains from the benign ones.

1) $|d|$ **, the length of a domain name**. Overall, DGA do- mains are longer than legal ones, and some domains generated by DGA have a fixed length.

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

2) v_ratio, c_ratio and *d_ratio*, **the ratio of the number of vowels, consonants and digits to domain length**. For the purpose of" easy to read", the proportion of vowels in legal domain names is relatively high while the DGA domains are the opposite. Besides, numeric characters are more common in

DGA domains. Figure 2 (*a*) ~ (*c*) illustrate this phenomenon with legal domains randomly selecetd from the Alexa Top List [14] and malicious ones generated by DGAs like *qakbot, necurs, ranbyus*, etc.

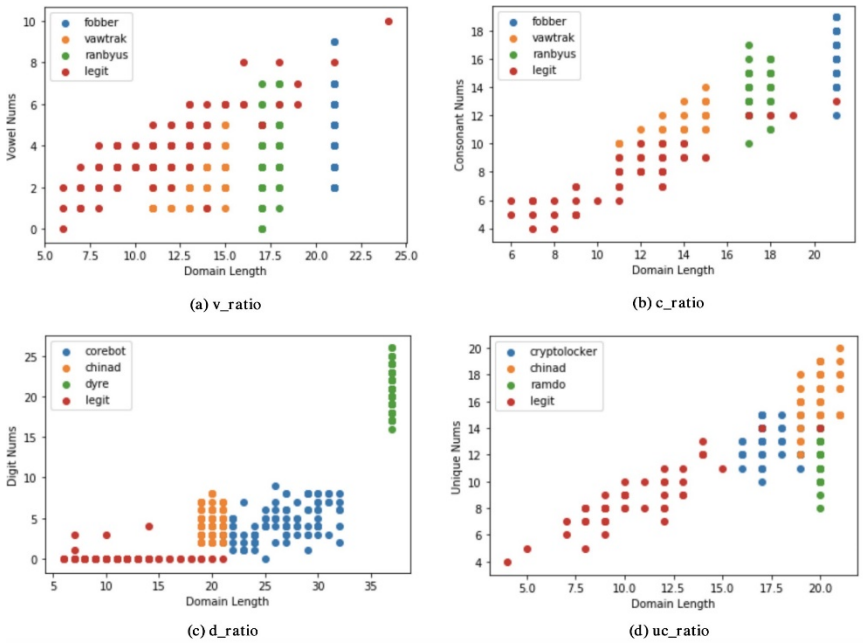

Figure. 2 Examples of the v_ratio, c_ratio, d_ratio and uc_ratio feature

3) uc_ratio, **the ratio of unique character counts to domain length**. As shown in Figure 2 (d), compared with DGA domains, the proportion of unique characters in legal domains is lower.

4) Ngrams, **statistical features based on Ngrams**. The basic idea of N-gram is to perform a sliding window operation of size N (N can be set as 1,2,3,4,5...) on the text, forming a sequence of fragments of length N. Each segment is called a gram, all grams together form a gram table, which is the vector feature space of this text. As the value of N increases, the dimension of the feature space will explode. Considering computing resources and computational complexity, 2-gram and 3-gram are often used in various tasks. Take *'google'* as an example, the character combination sets after 2-gram and 3-gram are *Set(2, google) = {go, oo, og, gl, le} and Set(3, google) = {goo, oog, ogl, gle}*.

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

In this project, we set N to be [2, 3] and extract 6 features: *ngram_max*, *ngram_mean*, *b ngram*, *m_ngram*, *bn- gram_ratio* and *mngram_ratio*. Specifically, let *Set(d)* in- dicates the segment collection of domain *d* after N-gram cutting and *Count (d, D)* represents the occurrences of *Set(d)*'s elements in collection *D*. *ngram_max* and *ngram _mean* are the max and mean value of *Count(d, total)*, where *total* means the whole domain name set. *b_ngram* and *m_ ngram* are *Count(d, benign) / Count(d, total)* and *Count(d, malicious) / Count(d, total)*, where *benign* is the legal domain set and *malicious* represents the DGA domain set. Besides, we use *bngram_ratio* and *mngram_ratio* to represent the DGA ratio and legal ratio of a domain *d*, which is *Count(d, benign) / |Set(d)|* and *Count (d, malicious) / |Set(d)|*. Figure 3 (a) ~ (c) shows the distribution of n-gram based features of legal domains and some DGA domains. We can see that legal domains tend to have higher *ngram mean* values, higher *bngram_ratio* and lower *mgram* values.

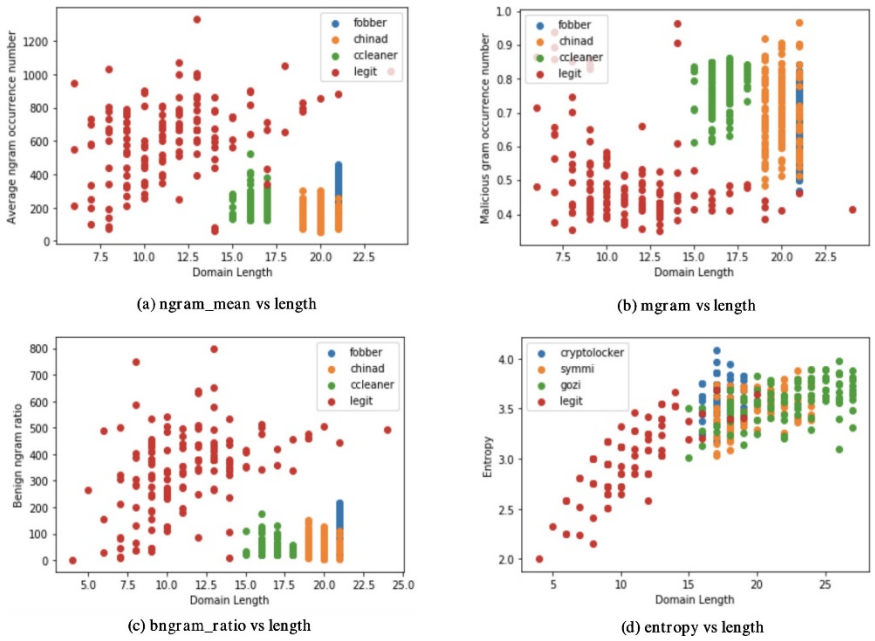

Figure. 3 Examples of the ngram_mean, mngram, bngram_ratio *and* entropy *features vs domain name length*

5) Entropy, **the Shannon entropy of a given domain**. The Shannon entropy $H = \sum_{i=1}^{n} p_i log p_i$, where p_i is the probability that character c_i appears in domain *d*, can show the randomness of character distribution. Figure 3 (d) illustrates that entropy of legal domains are much lower than those of DGA domains.

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

3.2. Detection Methods

In this project, we choose several basic machine learning methods and some ensemble methods to deal with the afore- mentioned 12 features and detect DGA domains.

1) Basic Machine Learning Methods: **Logistic Regression (LR)** is a classic ML method for binary classification. It assumes that the dependent variable *y* obeys Bernoulli distribu- tion, and introduces nonlinear factors with a *sigmoid* function. **Naive Bayes (NB)** is a classification method based on Bayes' theorem and the conditional independence hypothesis. The al- gorithm is relatively simple with few estimated parameters and is not sensitive to missing data. Theoretically, the NB model has a very small error rate. However, it cannot achieve the ideal performance as the conditional independence assumption is often invalid in practical applications. **Decision Tree (DT)** uses a tree structure and layers of reasoning to make the final classification. It contains the whole dataset in the root node and holds a certain attribute for judgment at each internal node. These attributes will help to determine which branch to choose until reaching a leaf node where the classification result is obtained. **Support Vector Machine (SVM)** tries to find the hyperplane with largest margin to separate the labeled dataset. Besides, it introduces nonlinearity through kernel functions.

2) Ensemble Methods: Ensemble learning methods try to get better performance by constituting a team of basic learners or models [12]. These methods can be divided into Voting, Bagging and Boosting. Voting is the simplest, as each model's prediction is regarded as a "vote" and prediction obtained by most models will be set as the final result. In this project, we combine the LR, NB, DT, SVM models into a VoteClassifier. With bagging methods, multiple models will be built based on different subsamples of the training data. Random Forest (RF) is a common bagging method where decision trees are independent of each other and the model can be fitted in parallel. On the contrary, there is a dependency between each basic learner on a boosting method. In each iteration, the misclassified learners will be given more weight and the errors will be rectified in the next time. XGBoost is one of the representative algorithms of boosting.

4. Experiments

In this section, we evaluate performance of the aforemen-tioned detection methods. We first introduce the composition of the training and testing dataset. Then, the experiment environment and evaluation metrics are listed in detail. Finally, we present the detection results.

The evaluation experiments are based on a publicly available dataset named UMUDGA [13]. Published on 24, Feb, 2020, this dataset presents a collection of over 30 million manually-labelled algorithmically generated domains extracted from 37 most important malware and their variants. Besides, it regards the Top 1M domain name from the Alexa list [14], a website ranking domains by visitor volume over a period of time, as the benign set. In this paper, we construct the malicious dataset by randomly selelcting GROUP_SIZE

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

domains from each DGA family in UMUDGA and take FAMILY_NUMS * GROUP_SIZE domains from Alexa as the benign set. Specifically, FAMILY NUMS = 37 and GROUP SIZE = 500 in our case.

Table 1 Metrics for evaluation

Dataset	Size	Resource
Benign	18500	Alexa Top List
Malicious	18500	37 dga families: cryptolocker,shiotob,pushdo,locky,alureon, chinad,gozi,symmi,dircrypt,vawtrak,padcrypt, rovnix,banjori,dyre,sisron,qadars,necurs, tempedreve,ccleaner,kraken,tinba,pykspa,qakbot suppobox,proslikefan,nymaim,pizd,bedep, corebot,zeus newgoz,simda,murofet,ramnit, fobber,ramdo,matsnu,ranbyus

The correctness of detection results can fall into four categories: True Positive (TP), which means DGA domains are labeled as malicious; True Negative (TN), indicating normal domains labeled as benign; False Positive (FP), indicating normal domains labeled as malicious, and False Negative (TN), which means DGA domains are labeled as benign. Four fundamental evaluation metrics (accuracy, precision, recall and F1 score) are calculated as follows. We can see that, accuracy is the ratio of correctly classified domains to the total observations, precision is the ratio of correctly detected DGA domains to the total detected samples, recall is the ratio of correctly detected DGA domains to true DGA domains and F1 score is a weighted average of precision and recall.

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

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

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

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

Besides, PR curve, ROC and AUC can provide new per- spectives for model comparison. A PR curve is a graph with precision values on ordinate and recall values on abscissa while the a ROC summarizes the trade-off between TPR and FPR with different thresholds. Both curves provide us a quick performance comparison

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

among different models. Specially, ROC is more suitable for the case where data is more evenly distributed, while PR curve is better at dealing with unbalanced datasets. AUC is defined as the area under a ROC enclosed by the coordinate axis. The closer the AUC is to 1.0, the better the performance of the detection method; when AUC is equal to 0.5, the method is of no application value.

In this project, the DGA detection system is implemented in Python 3.7 with third-party libraries like scikit-learn, pandas and auto-sklearn, etc. The experiments are deployed on a server with Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz, Tesla V100-PCIE-32GB, Ubuntu 16.04 LTS. For each experi- ment, we randomly separate the dataset into a training set and a testing set at the ratio of 7:3. In order to make full use of the effective information in the dataset and remit overfitting, we use 5-fold cross-validation in the experiments and the results shown below are all average statistics.

5. Results and discussions

Table 2 and Figure 4 illustrate the performance of afore- mentioned ML-based DGA classifiers. We can see that the ensemble learning-based classifiers are generally better than those based on a single ML model. Besides, the classifier with XGBoost surpasses the others with accuracy: 0.9030, preci- sion: 0.9268, recall: 0.8745 and shorter training time. What's more, the SVM-based classifier and the voting classifier with a SVM model in it has the longest and second-longest training time respectively. This is because the *svm.SVC* function in sklearn is implemented based on the libsvm library and has a $O(N^2)$ time complexity, which is not suitable for large-scale datasets.

Table 2 Detection results of different ML-based classifiers

Classifier	Accuracy	Precision	Recall	F1-score
LG	0.8617	0.8686	0.8524	0.8605
NB	0.8148	0.8781	0.7314	0.7981
SVM	0.8352	0.8420	0.8256	0.8337
DT	0.8592	0.8571	0.8625	0.8598
Voting	0.8716	0.8917	0.8442	0.8673
RF	0.9027	0.9303	0.8694	0.8989
XGBoost	0.9030	0.9268	0.8745	0.8999

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

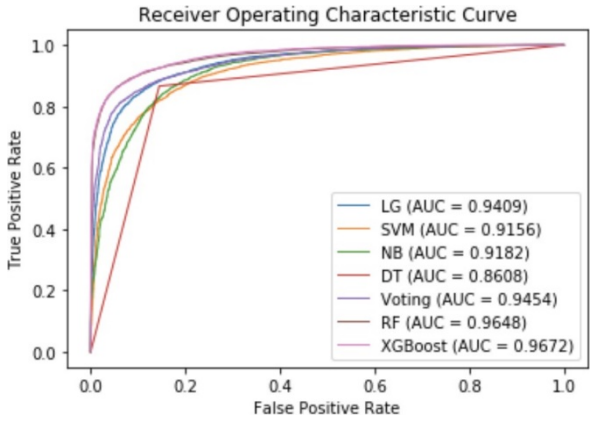

Figure. 4 ROCs of 7 different DGA classifiers

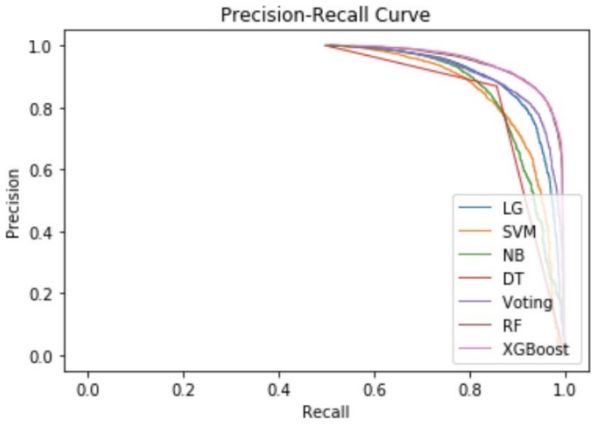

Figure. 5 PR curves of 7 different DGA classifiers

6. Conclusion

In this paper, we design an automatic DGA domain de- tection system and evaluate the detection performance among several machine learning methods on a public domain name dataset. With 12 manually selected features and an ensemble learning-based classifier, we can distinguish DGA domains from the legal ones with accuracy: 0.9030, precision: 0.9268 and recall 0.8745. We will investigate more effective features for more accurate detection and explore a more universally applicable ensemble learning method in the future.

ISSN 2616-5775 Vol. 3, Issue 4: 31-40, DOI: 10.25236/AJCIS.030404

References

[1] B. Fang, C. Xiang, and A. Wang, “Survey of botnets,” Journal of Computer Research Development, 2011.

[2] P. Porras, “inside risks reflections on conficker,” Communications of the Acm, vol. 52, no. 10, pp. 23–24, 2009.

[3] Sandeep, Yadav, Ashwath, Kumar, Krishna, Reddy, A., L., Narasimha, and R. and, “Detecting algorithmically generated domain-flux attacks with dns traffic analysis,” IEEE/ACM Transactions on Networking (TON), 2012.

[4] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, “Your botnet is my botnet: Analysis of a botnet takeover,” in Proceedings of the 16th ACM Conference on Computer and Communications Security, ser. CCS ’09, 2009, p. 635–647.

[5] Y. Zhauniarovich, I. Khalil, T. Yu, and M. Dacier, “A survey on ma- licious domains detection through dns data analysis,” ACM Computing Surveys (CSUR), vol. 51, no. 4, pp. 1–36, 2018.

[6] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. 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), 2012, pp. 491–506.

[7] 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 (TISSEC), vol. 16, no. 4, pp. 1–28, 2014.

[8] D. Plohmann, K. Yakdan, M. Klatt, J. Bader, and E. Gerhards-Padilla, “A comprehensive measurement study of domain generating malware,” in 25th USENIX Security Symposium (USENIX Security 16), 2016, pp. 263–278.

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

[10] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, and D. Dagon, “Detecting malware domains at the upper dns hierarchy.” in USENIX security symposium, vol. 11, 2011, pp. 1–16.

[11] S. Schu¨ppen, D. Teubert, P. Herrmann, and U. Meyer, “FANCI: Feature-based automated nxdomain classification and intelligence,”in 27th USENIX Security Symposium (USENIX Security 18), 2018, pp. 1165–1181.

[12] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, “A survey on ensemble learning,” Frontiers of Computer ence, vol. 14, no. 2, pp. 241–258, 2020.

[13] M. Zago, M. G. Pe´rez, and G. M. Pe´rez, “Umudga: a dataset for profiling dga-based botnet,” Computers & Security, vol. 92, p. 101719, 2020.

[14] 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].