Explaining Machine Learning DGA Detectors from DNS Traffic Data

Giorgio Piras[1], Maura Pintor[1,2], Luca Demetrio[1,2] and Battista Biggio[1,2]

[1]University of Cagliari, Cagliari, Italy
[2]Pluribus One S.r.l., Cagliari, Italy

Abstract

One of the most common causes of lack of continuity of online systems stems from a widely popular Cyber Attack known as Distributed Denial of Service (DDoS), in which a network of infected devices (botnet) gets exploited to flood the computational capacity of services through the commands of an attacker. This attack is made by leveraging the Domain Name System (DNS) technology through Domain Generation Algorithms (DGAs), a stealthy connection strategy that yet leaves suspicious data patterns. To detect such threats, advances in their analysis have been made. For the majority, they found Machine Learning (ML) as a solution, which can be highly effective in analyzing and classifying massive amounts of data. Although strongly performing, ML models have a certain degree of obscurity in their decision-making process. To cope with this problem, a branch of ML known as Explainable ML tries to break down the black-box nature of classifiers and make them interpretable and human-readable. This work addresses the problem of Explainable ML in the context of botnet and DGA detection, which at the best of our knowledge, is the first to concretely break down the decisions of ML classifiers when devised for botnet/DGA detection, therefore providing global and local explanations.

Keywords

Machine Learning, Explainability, Cybersecurity, DNS, Network Security, Monitoring and Detection

1. Introduction

During the last decades, our day-by-day life has been strictly connected to the usage of devices and online services, therefore making their efficiency and continuity play a crucial role in the technological transformation we witness. Likewise, the economic loss derived from cyber-threats has increased exponentially in recent years [1] as the technologies continually evolve and attackers develop their skills. One of the most common ways cybercriminals try to jeopardize the continuity of systems and thus cause economic damage is Denial of Service (DoS), which aims to drain the computing capabilities of the target system in both fancy and basic ways. A case of this attack is the Distributed Denial of Service DDoS, where a network of infected devices (bots) are commanded by an attacker (botmaster) through a Command&Control Server (C&C) [2, 3, 4]. What happens to be erratic and thus detectable by a Machine Learning (ML) model in this kind of attack is the DNS traffic, carrying Domain Names through which bots are

ITASEC'22: Italian Conference on Cybersecurity, June 20–23, 2022, Rome, Italy

giorgio.piras@unica.it (G. Piras); maura.pintor@unica.it (M. Pintor); luca.demetrio93@unica.it (L. Demetrio); battista.biggio@unica.it (B. Biggio)
0000-0001-8225-6138 (G. Piras); 0000-0002-1944-2875 (M. Pintor); 0000-0001-5104-1476 (L. Demetrio); 0000-0001-7752-509X (B. Biggio)

CEUR Workshop Proceedings (CEUR-WS.org)

connected to the C&C server. This stealthy connection strategy is commonly known as Domain Fluxing, where the algorithms used by the infected bots to generate the domain are known as Domain Generation Algorithms (DGAs).

Although employing ML models to detect the presence of botnets within network traffic has been demonstrated to be successful, almost the entirety of the relevant works have followed a common baseline and workflow, presenting a partially novel feature set on which to train a classifier to obtain relevant results [5, 6, 7, 8]. The proposed approaches lack interpretability and contextualization. First, depending on the context from which DNS traffic data is extracted and the model is deployed, potential attackers might have control over some features. Second, the model prioritization and general usage of the features in the decision process are not known beforehand, making the process challenging to debug and protect.

To make up for these problems, we first analyze the techniques used to detect botnets/DGAs from the DNS data (Section 2); we analyze which explainability techniques can provide insight into how the model takes its decisions (Section 3). Upon a re-implementation of the EXPOSURE system [9, 5] (Section 4), we provide the following contributions: (i) we build and test the EXPOSURE system on a newly collected dataset; (ii) we observe statistics on the features used by the system; (iii) we train different classifiers and compare their performances; (iv) we obtain explanations from such classifiers; and (v) given the explanations, we develop and discuss an analysis on the features used by the systems mentioned above. Finally, we conclude the work by presenting related works (Section 5), limitations, and future directions (Section 6).

2. Background: DNS System and ML Techniques

From DNS to DGA. The Domain Name System (DNS) is a database responsible for mapping domain names to IP addresses, thus answering a query made by clients in the form of a domain name towards the IP addresses. This action is commonly known as *resolution* [10]. The DNS organizes domain names into a hierarchy (through dot-separated levels), as the whole technology itself creates a hierarchical database structure. The information stored and carried by DNS records can be A records, returning IPv4 addresses, NS records, returning authoritative name servers, and finally, PTR, which stands for Pointer to Record and returns a domain name but in the reverse query format (i.e., the question started from an IP rather than a domain name). It is also worth citing specific information carried by the DNS packets, such as the Time-to-live (TTL), which indicates how long the server will cache that packet [11, 10]. Being of paramount importance for the correct functioning of basic internet activities, the DNS is the perfect target for malicious activities having a high impact on unaware users. That is why this technology gets exploited by attackers (*botmasters*) who aim to command and control a network of infected machines, i.e., a botnet. To go as undercover as possible, having only one domain name to which to connect would have the botnet quickly taken down by vigilant authorities. That is why bots generate massive DNS traffic trying to connect to a much more concealed C&C server. The generation of such a significant amount of domain names happens through Domain Generation Algorithms that, given a random seed, create a string that will possibly establish a connection.

Botnet Detection with ML: The EXPOSURE system. Seizing the chance to detect malicious patterns, the research community has driven its efforts towards analyzing the DNS data, ex-

Table 1
List of features used in EXPOSURE renewed with a mnemonic. The first column indicates the Feature subset. The second one shows the number of features that a specific feature holds. Finally, the third and fourth columns indicate respectively the feature name chosen by the authors and our feature names for atomic features.

Feature set	#	Paper Feature Name	Our Feature Names
Time-Based Features	2	Short Life	glob_short_lived glob_life_ratio
	1	Daily similarity	daily_similarity
	2	Repeating patterns	local_numOf_changes stddev_before_change
	2	Access Ratio	idle popular
DNS Answer-Based Features	1	Number of distinct IP addresses	unique_ips
	1	Number of distinct countries	unique_ccode
	3	Reverse DNS query result	rev_arec rev_nsrec rev_asnrec
	1	Number of domains sharing the same IP	shared_ips
TTL Value-Based Features	1	Average TTL	ttl_avg
	1	Standard Deviation of TTL	ttl_stddev
	1	Number of distinct TTL values	unique_ttls
	1	Number of TTL changes	ttl_changes
	5	Percentage usage of TTL ranges	ttl_range1 ttl_range100 ttl_range300 ttl_range900 ttl_rangeinf
Domain Name-Based Features	1	% of numerical characters	num_chars%
	1	% of length of the LMS	%of_lms

tracting the features, and eventually training a ML model capable to distinguish malicious and benign DNS behaviors. The EXPOSURE system [9, 5] is among the most prominent works for its completeness in the feature set and reproducibility (in terms of feature extraction). For this reason, we use it as a base for our explainability analysis. Table 1 shows the feature set, listing the features extracted by the EXPOSURE system (whose extensive description can be found in the original work [9, 5]). The entire set is subdivided into Time-Based features (collecting temporal patterns from the queries to the domains), DNS-Answer-Based Features (patterns from the answers records), TTL-Value-Based Features (statistical patterns from the TTL values), and finally, Name-Based Features (statistical patterns from the Domain Name). Given a collection of DNS packets, we can compose a training set of benign and malicious samples to train the classifier, as Bilge et al. did in EXPOSURE. Our work will focus on reproducing the experiment with our newly-collected traffic and applying explainability techniques to understand the patterns employed by the model for detecting malicious activities.

3. Explaining Predictions of ML-based DNS Analysis

As pointed out by Miller et al. [12], explanations increase transparency and interpretability so that user awareness and systems designers can jointly benefit from this gain of trust. In security-relevant scenarios, like the one we are considering, understanding the data and the model provides the added benefit of helping to see if there are problems in the system, for example assigning high relevance to spurious features that should not influence it to that extent [13].

Analyzing the dataset's statistics provides further insights into the separability of the features into the two different classes. Additionally, in the case under investigation, lots of features come from similar sources and elaborations, which lets the statistical analysis come in handy to highlight correlations and redundancies.

On top of that, we will use a ML model to analyze such features and categorize the samples into the two output classes of *benign* and *malicious* domains. Model explanations can help understand how the model is making such decisions. An explanation is said to be local if it is made on single samples and wants to describe how a model emphasizes the features of a specific single sample in its classification. On the other hand, global explanations are made over entire datasets or relevant collections of samples to describe how the model prioritizes features over those samples [14]. This work will focus on both local and global explanations.

In [15], Lundberg and Lee proposed SHAP (SHapley Additive exPlanations), where feature importance is computed with an additive approach, representing a unified measure of feature importance. The basic concept behind SHAP comes from Shapley values and a game theory setting, where the features act as players and cooperate in a coalitional game (i.e., the prediction task) to receive a profit (i.e., a gain, which is the actual prediction). The Shapley values assign payouts to players depending on their contribution to the total payout [16]. Thus each feature that contributes to the prediction task is computed as a sum of the expected marginal contributions in any feature value combination. Given the computational burden for which SHAP should find all the possible feature combinations, Lundberg and Lee proposed a Shapley kernel that produces estimates instead of exact values.

3.1. Data Analysis and Explainability Techniques

This section will briefly explain the details and differences between the data analysis and explainability tools that will play a central role in the experiments section.

Feature Statistical Analysis. These plots show the marginal distributions of every pair of features as density plots, describing how the distributions for the classes behave. Through the scattered plots instead, we can assess where both benign and malicious samples lie in their ad-hoc feature space, thus making us capable of understanding to which extent pairs of features separate the data. Analyzing the scattered plots allows observing the distribution of the features to get a rough idea of how they will behave/discriminate and to which extent.

Partial Dependence Plot. This plot shows the marginal effect that a single feature has on the prediction made by the model, thus providing global explanations. Taking as input the model, the feature, and a background distribution on which to make the model learn the feature importance, the Partial Dependence Plots (PDPs) depict the feature values on the x-axis, whilst the y-axis represents the expected prediction contribution given the feature value. In the background, a histogram shows the underlying data distribution of the feature values. A horizontal line represents the expected contribution to the prediction, and a vertical one represents the expected value of the feature. By reading this plot, we can measure how the observed feature contributes to the classification of the samples.

Summary Plot. The SHAP summary plot, which as the PDP is a global explanation technique, shows how the model prioritizes the features and how these contribute to steering the classification towards each class. This plot comprises a list of features ordered from the one giving the higher contribution to the least powerful as interpreted by the model, showing the magnitude for benign (in blue) and malicious (in red) samples.

Force Plot. This technique is one of the local explainability methods provided in SHAP. It explains why a specific sample has been assigned a particular label. This can be useful for understanding why samples are misclassified and to which extent the classifier misunderstands them. Force plots, showing the magnitude of the feature contribution on single samples, are rendered as blue arrows indicating magnitude values towards the benign class and red vice-versa.

In the next section, we will use the presented techniques to explain predictions of our re-implementation of the EXPOSURE system.

4. Experiments

In the experimental section of this work, we will first describe our re-implementation of the EXPOSURE system. Then we will discuss the DNS traffic data we used to make our feature extraction, followed by a brief model selection made to improve the system's performance. Eventually, we will delve into the results section to show how explanations applied in this context can bring the analysis to the next level.

4.1. Re-implementation of the EXPOSURE system

Dataset. The DNS traffic was collected from recursive servers on which, through sniffers, we were able to save the data as `.pcap` files for the entire month of January 2021. Given the massive amount of traffic, summing up to 15 GB of data per day, we filtered out packets whose label was not known by either black or white lists and domain names that did not resolve (`NXDOMAIN` as response code). We used the list of most popular suffixes from the Alexa website[1] to label benign domains, and the list from DGArchive [17] to flag the malicious samples. The remaining packets ($203,034$ domains, of which $25,882$ benign and $177,152$ malicious - note that benign domains re-appear much more frequently than the malicious ones, which are almost always unique) have then been passed through the feature extractor we implemented, and are distributed through days as shown in Figure 1.

Model Selection. The authors used a J48 Decision Tree to obtain overall good performances in the original EXPOSURE work. We additionally bench-marked several models such as Decision Tree (DT), K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Ada-Boost (ADA), and

[1] https://www.alexa.com/topsites

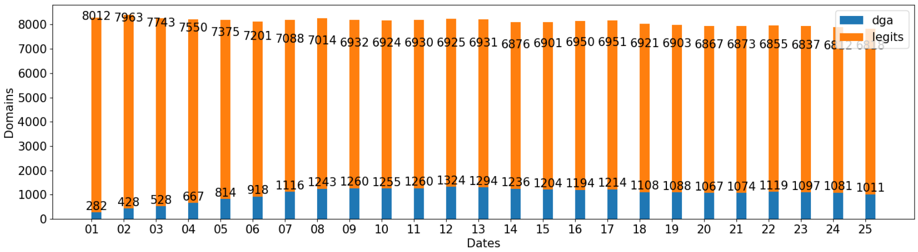
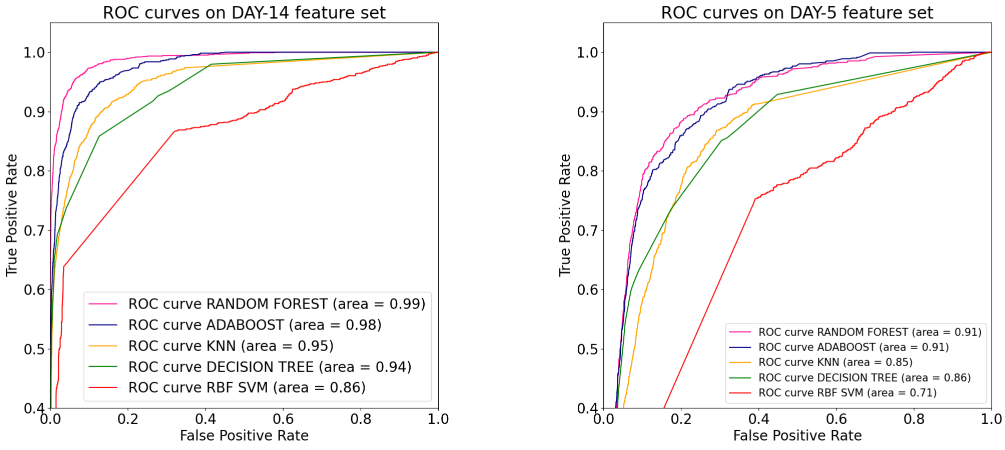


Figure 1: Malicious and legitimate domains for every day.

Random Forest (RF). After estimating the hyperparameters through a Grid Search, (whose bests overall have been reported in Section 7) we compared the best models with the best parameters on two different days (i.e., two different sample balances). The first ROC curve in Figure 2a was obtained using a more balanced day of data (mid of January days). The second set of curves in Figure 2b was obtained from a day of data with very few malicious samples, showing how the performances of the classifiers dropped down consistently. Overall, throughout the most balanced days, RF and ADA have shown to be the more consistent classifiers. For this reason, they have been selected as classifiers for the rest of the experiments. We reckon that the mid-days of capture are also more suitable for the rest of the analysis, and they have been thereby used for all of the following experiments.



(a) Classifiers trained on day 14, with class distribution unbalanced but less evident than the other days in the dataset.

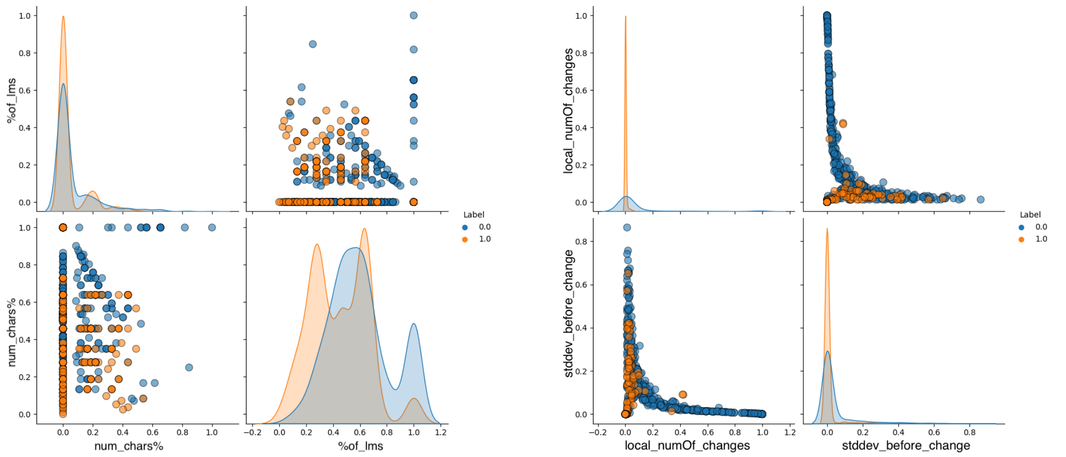
(b) The same classifiers trained on day 5, which presents a highly-unbalanced distribution of the two classes.

Figure 2: The ROC curves obtained using features from day 14 (*left*) and on day 5 (*right*).

4.2. Feature and Explainability Analysis on EXPOSURE

We now present our results on the analysis of the feature statistics and our insight obtained through applying explainability techniques to interpret the decisions taken by the machine-learning models used in our DGA detector. These proposed plots have been implemented through the Python libraries Seaborn [18] and SHAP [19].

Statistical Analysis. The statistical analysis shows an overview of the correlation and distribution of the features. As shown in Figure 3a, some features like the `%of_lms` and `num_chars%` when joined, do not separate perfectly the data collection used into the two classes. In particular, the `%of_lms` reaches a plateau in malicious domains once over 0.8 (bottom-right plot, depicting the distribution of the feature), which describes how algorithmically-generated domains tend not to have a single meaningful word covering their entire name in most of the cases, yet there are exceptions in any direction. This might be brought on by the diversity in malware families, where some like "Gameover" DGA used to mix up numbers and characters. In contrast, others such as "Gozi" used to mix up words from openly accessible documents, such as the US constitution [20]. In the plot of Figure 3b, depicting time-based features, malicious domains show a more volatile behavior, which is reasonable if we think about the diversity of applications in which they can be used. In Figure 8, shown in the Appendix, we can observe interesting TTL behaviors characterizing the domains. Contrary to the now old-fashioned belief that a low TTL is only typical for malicious domains [21, 22], as it makes malicious records stand less in caches, we show that also benign domains can present this behavior depending on the application in which they are used, e.g., to handle critical resources [23] or for load balancing purposes [22].

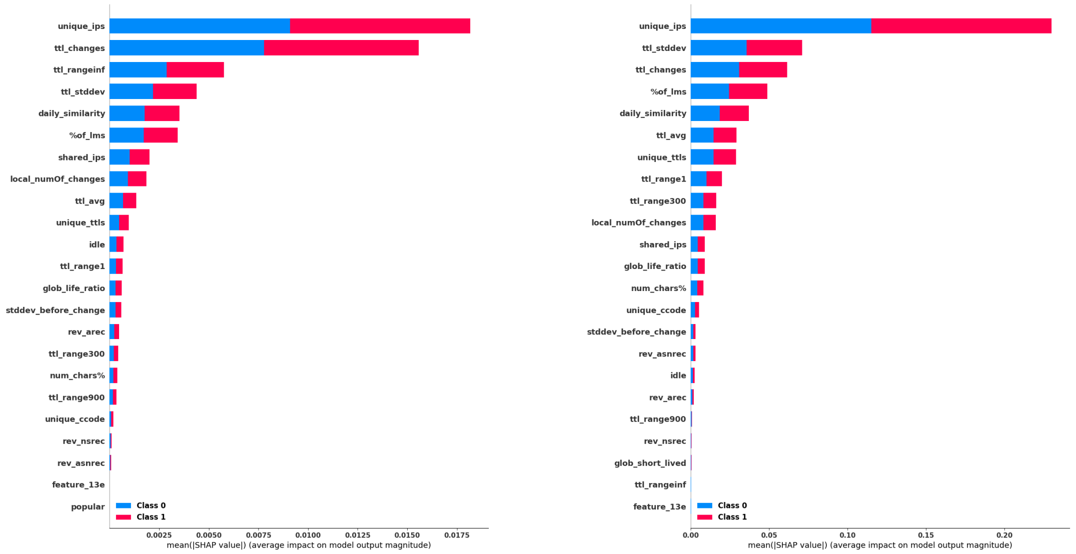


(a) Correlation between name-based features.

(b) Correlation between time-based features.

Figure 3: Density and scatter plots for two pairs of name-based (*left*) and time-based features (*right*). The plots are based on the 12th day of capture, which counts 1324 malicious samples and 1324 benign samples (resized from 6925).

Interpreting Global Explanations. As pointed out in Section 3, different models can use features in different ways. Global explanations can uncover these behaviors and let the analyst be aware of the feature prioritization that a model brings. Some Decision-Tree based classifiers such as ADA and RF, respectively in the summary plots of Figure 4a and Figure 4b, share four out of the five top important features, which is likely to be a consequence of their “similar” tree-based intrinsic nature. In both of them, `unique_ips` notably brings the higher contribution. In Section 7, we show how other classifiers have a low magnitude provided by the `unique_ips` feature, whilst prioritizing a diverse subset of TTL features. This furtherly shows how it is not possible to solely rely on statistical analysis to foresee the utilization of the features, as class separations that at first glance look either weak or strong can be subverted.

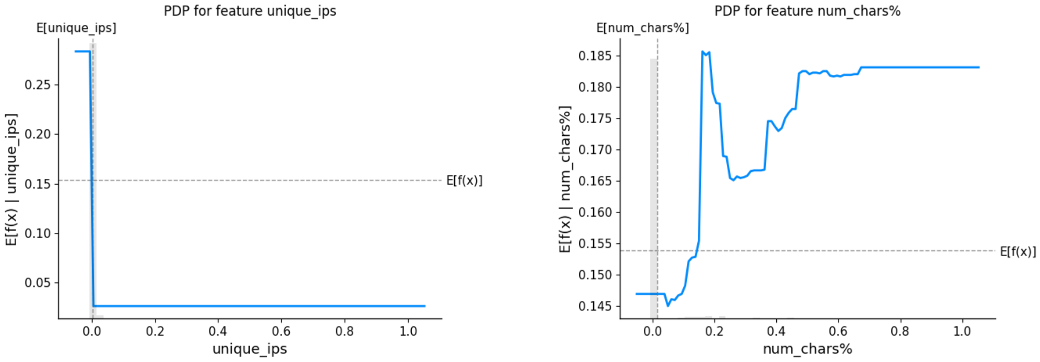


(a) SHAP summary plot of feature contributions on ADA BOOST classifier.

(b) SHAP summary plot of feature contributions on RANDOM FOREST classifier.

Figure 4: Global summary plots for ADA (*left*) and RF (*right*) classifiers.

Partial Dependence Plots show the marginal effect of a single feature globally on the predictions. Considering a trained classifier (RF in the case of this analysis) and a background distribution, through SHAP we can assess how the considered feature contributes to classifying the background samples over their values. An advisable security-related use of these plots can be to employ a background distribution of malicious samples, thus analyzing to which extent the feature values contribute to classifying the sample as malicious. The following plots (after normalization) have been made using a background distribution of 1000 malicious samples on the RF classifier. Figure 5a shows the PDP of the strongest feature of the model. The plot tends to be a “gentle” step, producing the highest contribution on very low feature values and the lowest with values going just subtly over the threshold. Very similar behavior to the one of the `unique_ips` feature is shown in the number of changes in the TTL, depicted in Figure 7 in Section 7. These plots help, for example, understanding the extent to which features contribute to the classification of the domain as malicious and possibly setting policies and restrictions based on simply tweakable features, such as the `num_chars%` in Figure 5b. In this plot, we can understand how a high rate of numerical characters leads to a solid contribution to the prediction of the domain as malicious. Likewise, it is surprising that a 20% rate of numerical characters in the domain string leads to an even bigger magnitude, which can be caused by the relevant presence of some malware families not having numbers in their “regex”. In this case, usage of the proposed security policies for a system hosting an EXPOSURE-like system would be to allow domain names with numerical characters comprised in between the 20%-40% range.



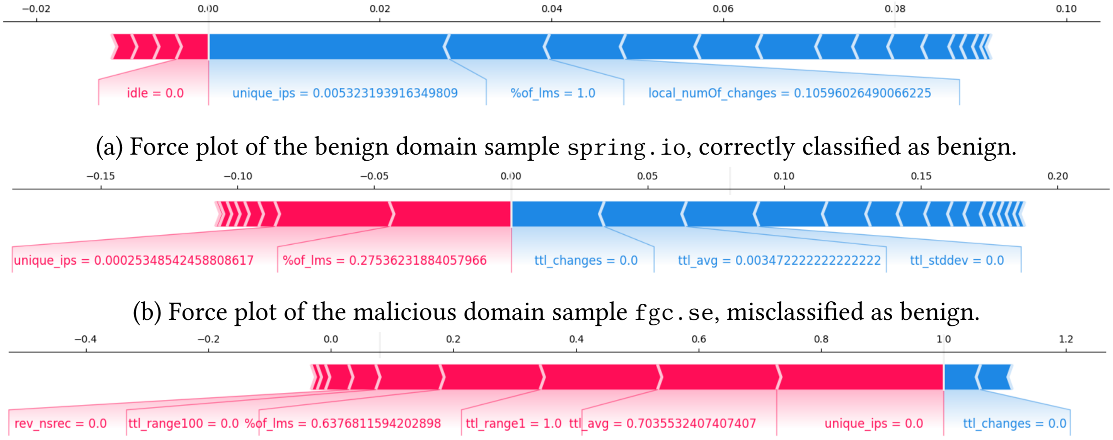
(a) SHAP PDP on `unique_ips` feature.

(b) SHAP PDP on `num_chars%` feature.

Figure 5: Partial Dependence plots for `unique_ips` (*left*) and `num_chars%` (*right*)

Magnifying the Behavior with Local Explanations. Force plots and local explanations can be considered the local version of summary plots, showing how features contribute locally to the sample. As Figure 6a shows, the features correctly lead the RF model classification of the domain `spring.io` as benign. It is not surprising to see a 0 value of `unique_ips` forcing the classification towards malicious, but the rest of the features gently move the prediction towards the *benign* class. For malicious samples, instead, the latter plays a major role incorrectly classifying some malicious samples like `mobile.de` and `qcx.nl`, as additionally shown in the Appendix. In general, the magnitude of the features sticks to what is shown by the summary plot of Figure 4b for samples correctly classified. To understand instead which features are leading the model to misclassify a sample, Figure 6b shows the force plot for the domain `fgc.es`, blacklisted but yet misclassified by our system as *benign*. The `unique_ips` and `%of_lms` features correctly move the prediction towards the *malicious* class, but the values of the TTL-based features deceive the classifier. Figure 6c, on the other hand, shows how `unique_ips` and TTL values deviate the prediction of the benign domain towards the malicious class.

Summary of the Results. As a result of the presented experiments, we can reflect on the issues of feature management and hypothetical counteractions. We understand the features distribution, correlation, and how the DGAs in our traffic tend to behave from statistical analysis. However, the global explanations can turn the table and quantify how the model perceives the features. Finally, we can see how features drive the sample’s prediction via local explanations. This ensemble of analysis makes us notice how the overall feature prioritization depends on both the model used and the considered data, which further proves how context-dependent such systems’ behavior can be. Hence, an Explainability analysis should always be used to better portray the big picture of both systems and employed data. In our case, the big picture



(a) Force plot of the benign domain sample `spring.io`, correctly classified as benign.

(b) Force plot of the malicious domain sample `fgc.se`, misclassified as benign.

(c) Force plot of the benign domain sample `topeleven.com`, misclassified as malicious.

Figure 6: Local explanations on three domains from the dataset.

has led to a more prominent analysis of the EXPOSURE system. The TTL features, the subject of this analysis, sum up to 37.5% of the entire feature set, being 9 out of 24. It turns out from our analysis of the explanations that they contribute massively to the misclassification of several samples (such as Figure 6b and Figure 6c) as they cover high-magnitude roles in the summary plots of Figure 4a and Figure 4b (which are the best classifiers overall). This makes this feature set highly powerful for the whole system, yet in its power also lies a crucial problem. Namely, attackers can manipulate this feature, being completely free to tune the TTL and balance their caching time (i.e., the likelihood of being detected) with the chance of evading the classification of the system. Having such a relevant portion of the feature set reserved for values that can be somehow crafted directly by the attackers, it can serve as a significant stepping-stone for attackers. Furthermore, if deployed on a system devised to manage critical resources, besides evading, some part of the feature set can be overridden by the context. Some works like [23] point out how security-sensitive systems, e.g., banking applications, should indeed carefully set their DNS TTL to a low value. The scope of these assumptions is that a botnet/DGA detector cannot solely rely on accuracy metrics to establish its efficiency, in that analysts also need to be aware of the model, the data, and the context therein. Explanations can give a huge and crucial hand in this regard, helping prevent major issues from happening and allowing debugging of the model. Considering our system, through explanations, we have seen how dangerously influential TTL-based features are in most of the models. And considering their extensive use in the feature set, appropriate security measures should be taken (e.g., reducing their number like for Name-Based features, which are just easily adjustable as well but sum up to only the 8%). We firmly believe that through explanations we can rapidly enhance the usage and trust in AI, as companies can look at such security systems from a human-readable perspective, and model biases can be analyzed and studied.

5. Related Work

DNS Analysis. Several promising works have striven to tackle botnet/DGA detection during the last decade, often showing innovative DNS passive features and methodologies. Some notable works, besides EXPOSURE, have been Notos [24], where Antonakakis et al. created the first relevant and efficient reputation system for domains from various data sources. Pleiades [25], where the authors focused on NXDOMAIN records to both cluster domains and classify DGAs by looking at the strings association. Finally, in FANCI [6], Schueppen et al. developed a detector based on a small feature set such as the EXPOSURE one, though focusing only on NXDOMAIN passive data. All of these works have reached comparable performances in different settings. None of them, though, has focused their interest on the explainability of such a critical application. The only works to have addressed such problems focused on multiclass classification problems with deep learning approaches, thus classifying the malicious domains with family pairing. In [26], Becker et al. proposed a visual analytics system for Deep Learning (DL) models, providing graphical insights on statistical properties of the domain name string. Drichel et al., in [27], briefly highlighted some string-wise interpretations for DL models starting from the misclassified samples. In contrast, in [28], Drichel et al. proposed feature-based classifiers based on string features for multiclass classification, with the purpose of improving explainability. Firstly, none of these three works focused on passive DNS data, choosing string-based features to ease the computational burden. Secondly, none developed explanatory analysis, rather focusing on how the model could be made more explainable or at most on how to visualize a few string patterns from a DL model. Our work focuses on passive DNS traffic data, analyzing features from a comprehensive viewpoint and not limiting them to the human-readable string features. Additionally, we propose both local and global explanations, concretely enhancing the awareness of how a model behaves in such a context.

Explainability Techniques. In [29], Ribeiro et al. proposed LIME (Local Interpretable Model-Agnostic Explanations), an explainability method conceived as a local model learning and approximating around the prediction. Despite its wide use, several concerns about stability and consistency have been addressed towards LIME [15, 30]. Considering that SHAP is a more reliable tool, we have driven our choice towards its use in our explainability work.

6. Conclusions and Future Work

In this work, we proposed an explanatory analysis of ML classifiers devised for botnet/DGA detection. Starting from the implementation of the EXPOSURE feature set on our traffic data, we have shown how from prior statistical assumptions on the malware behavior within the network, a model can interpret features in its way globally, thus prioritizing certain features rather than others that were prevented, also demonstrating how different models can have a different feature conception, to which eventually we analysts should adapt and debug accordingly. Locally, we have seen how certain features can contribute and how explanations can make the analysts and users aware of the single decisions and motivations behind misclassified samples. Through these analyses, we raised concerns about how the feature and model can be biased by the context in which the systems are both trained and deployed. And our analysis makes the comprehension of such contexts move fast forward towards favoring the employment of such systems, as they can be firstly interpreted and adapted and subsequently accepted. In this regard, several advances of this work can be developed aiming at fairness and legal regularization of the detectors through explanations and, if possible, bringing them into debugging/pipelining processes to obtain an efficient and explainable system. Additionally, they can be instrumental when humans want to be involved in the decision-making. All in all, this work demonstrated how powerful explanations can be and how security, debugging, interpretability, and fairness can be brought to the next level by the application of ML to detection, where security has to be assessed and interpreted through the process chain.

Acknowledgments

This work has been partly supported by the PRIN 2017 project RexLearn, funded by the Italian Ministry of Education, University and Research (grant no. 2017TWNMH2); and by the project TESTABLE (grant no. 101019206), under the EU's H2020 research and innovation programme.

References

[1] Z. Smith, E. Lostri, M. (Firm), The Hidden Costs of Cybercrime, McAfee, 2020. URL: https://books.google.it/books?id=mG0jzgEACAAJ.

[2] R. Puri, Bots &; Botnet: An Overview, Elsevier (2003) 17.

[3] W. Salusky, R. Danford, Know Your Enemy: Fast-Flux Service Networks, in: The Honeypot Project, 2007.

[4] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, G. Vigna, Your botnet is my botnet: analysis of a botnet takeover, in: Proceedings of the 16th ACM conference on Computer and communications security - CCS '09, ACM Press, Chicago, Illinois, USA, 2009, p. 635. URL: http://portal.acm.org/citation.cfm?doid=1653662.1653738. doi:10.1145/1653662.1653738.

[5] L. Bilge, E. Kirda, C. Kruegel, M. Balduzzi, Exposure: Finding malicious domains using passive dns analysis., in: Network and Distributed System Security Symposium (NDSS), 2011.

[6] S. Schüppen, D. Teubert, P. Herrmann, U. Meyer, FANCI : Feature-based automated nxdomain classification and intelligence, in: 27th USENIX Security Symposium (USENIX Security 18), USENIX Association, Baltimore, MD, 2018, pp. 1165–1181. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/schuppen.

[7] C. Zheng, Q. Qiang, T. Zang, W. Chao, Y. Zhou, Themis: A Novel Detection Approach for Detecting Mixed Algorithmically Generated Domains, in: 2019 15th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), 2019, pp. 259–264. doi:10.1109/MSN48538.2019.00057.

[8] S. Schiavoni, F. Maggi, L. Cavallaro, S. Zanero, Phoenix: Dga-based botnet tracking and intelligence, in: Detection of intrusions and malware, and vulnerability assessment, Springer, 2014, pp. 192–211.

[9] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, C. Kruegel, Exposure: A passive dns analysis

service to detect and report malicious domains, ACM Trans. Inf. Syst. Secur. 16 (2014). URL: https://doi.org/10.1145/2584679. doi:10.1145/2584679.

[10] P. V. Mockapetris, Domain names - implementation and specification, ????. URL: https://tools.ietf.org/html/rfc1035.

[11] P. V. Mockapetris, Domain names - concepts and facilities, ????. URL: https://tools.ietf.org/html/rfc1034.

[12] T. Miller, Explanation in Artificial Intelligence: Insights from the Social Sciences, arXiv:1706.07269 [cs] (2018). URL: http://arxiv.org/abs/1706.07269, arXiv: 1706.07269.

[13] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, K. Rieck, Dos and don'ts of machine learning in computer security, in: Proceedings of the 31st USENIX Security Symposium, 2020.

[14] C. Molnar, 2.5 Properties of Explanations | Interpretable Machine Learning, https://christophm.github.io/interpretable-ml-book/, 2019. URL: https://christophm.github.io/interpretable-ml-book/properties.html.

[15] S. Lundberg, S.-I. Lee, A Unified Approach to Interpreting Model Predictions, arXiv:1705.07874 [cs, stat] (2017). URL: http://arxiv.org/abs/1705.07874, arXiv: 1705.07874.

[16] S. Hart, Shapley Value, in: The New Palgrave Dictionary of Economics, Palgrave Macmillan UK, London, 2017, pp. 1–5. URL: https://doi.org/10.1057/978-1-349-95121-5_1369-2. doi:10.1057/978-1-349-95121-5_1369-2.

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

[18] M. L. Waskom, seaborn: statistical data visualization, Journal of Open Source Software 6 (2021) 3021. URL: https://doi.org/10.21105/joss.03021. doi:10.21105/joss.03021.

[19] An introduction to explainable AI with Shapley values — SHAP latest documentation, ???? URL: https://shap.readthedocs.io/en/latest/example_notebooks/overviews/An%20introduction%20to%20explainable%20AI%20with%20Shapley%20values.html.

[20] F. F. Daniel Plohmann, Dgarchive, https://dgarchive.caad.fkie.fraunhofer.de, ????

[21] R. Villamarin-Salomon, J. C. Brustoloni, Identifying botnets using anomaly detection techniques applied to dns traffic, in: 2008 5th IEEE Consumer Communications and Networking Conference, 2008, pp. 476–481. doi:10.1109/ccnc08.2007.112.

[22] K. Alieyan, A. ALmomani, A. Manasrah, M. M. Kadhum, A survey of botnet detection based on DNS, Neural Computing and Applications 28 (2017) 1541–1558. URL: http://link.springer.com/10.1007/s00521-015-2128-0. doi:10.1007/s00521-015-2128-0.

[23] N. Vlajic, M. Andrade, U. T. Nguyen, The Role of DNS TTL Values in Potential DDoS Attacks: What Do the Major Banks Know About It?, Procedia Computer Science 10 (2012) 466–473. URL: https://www.sciencedirect.com/science/article/pii/S1877050912004176. doi:10.1016/j.procs.2012.06.060.

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

[25] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, 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.

[26] F. Becker, A. Drichel, C. Müller, T. Ertl, Interpretable visualizations of deep neural networks

for domain generation algorithm detection, in: 2020 IEEE Symposium on Visualization for Cyber Security (VizSec), 2020, pp. 25–29. doi:10.1109/VizSec51108.2020.00010.

[27] A. Drichel, U. Meyer, S. Schüppen, D. Teubert, Analyzing the real-world applicability of dga classifiers, in: Proceedings of the 15th International Conference on Availability, Reliability and Security, 2020, pp. 1–11. doi:10.1145/3407023.3407030.

[28] A. Drichel, N. Faerber, U. Meyer, First step towards explainable dga multiclass classification, in: Proceedings of the 16th International Conference on Availability, Reliability and Security, 2021, pp. 1–13. doi:10.1145/3465481.3465749.

[29] M. T. Ribeiro, S. Singh, C. Guestrin, ”Why Should I Trust You?”: Explaining the Predictions of Any Classifier, arXiv:1602.04938 [cs, stat] (2016). URL: http://arxiv.org/abs/1602.04938, arXiv: 1602.04938 version: 1.

[30] Z. Zhou, G. Hooker, F. Wang, S-LIME: Stabilized-LIME for Model Explanation, Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (2021) 2429–2438. URL: http://arxiv.org/abs/2106.07875. doi:10.1145/3447548.3467274, arXiv: 2106.07875.

7. Appendix

In this additional section, we show several plots that have been cited in the previous sections and that we believe can support the comprehension of the work.

Grid Search Results. Using the Scikit-Learn Python suite, we optimized the parameters through the GridSearchCV API. The results of the optimization have been reported for completeness in Listing 1.

TTL Features plots. Having focused the discussion of the explainability analysis almost entirely on the TTL features, there are some additional plots that can point out interesting behaviors, such as Figure 8, which shows the statistical analysis of the first 4 TTL-based features. In Figure 7 instead, we can see how low changes in the TTL values mean a low contribution to the classification of the sample as malicious and vice versa.

Additional Summary Plots. Figure 9 shows how `unique_ips` are much less considered than the TTL-based features by the KNN classifier, which again shows how models are as diverse as they are. The same goes for the SVC classifier in Figure 10, which once again does not employ the `unique_ips` feature as much as the Decision-Tree based classifiers do.

Additional Force Plots. The plots of Figure 11 show a variety of samples either correctly classified or misclassified by the RF model, demonstrating practically how the most relevant features can play a major role in any classification scenario, either in the wrong or correct way.

Listing 1: Grid Search Results

```
# RANDOM FOREST parameters
{'criterion': 'entropy', 'max_depth': 20, 'n_estimators': 125}
# ADA-BOOST parameters
{'algorithm': 'SAMME', 'n_estimators': 175}
# K-NEAREST NEIGHBORS parameters
{'n_neighbors': 13, 'weights': 'distance'}
# DECISION TREE parameters
{'criterion': 'gini', 'max_depth': 5}
# SVC RBF parameters
{'C': 297.63514416313194, 'gamma': 0.6951927961775591}
```

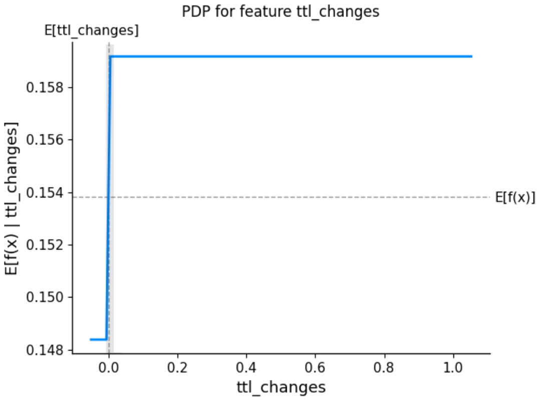


Figure 7: SHAP PDP on *ttl_changes* feature.

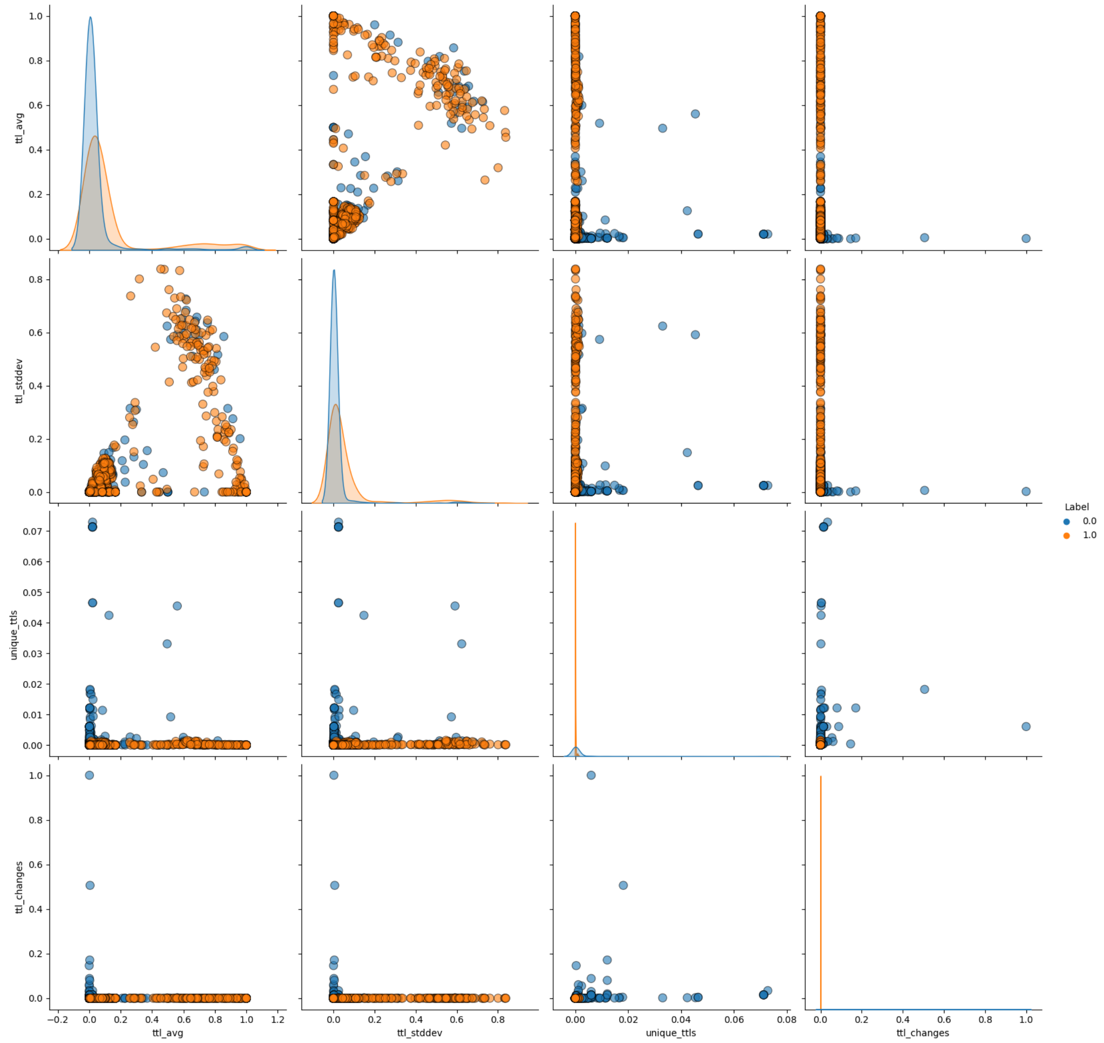


Figure 8: Correlation between the first four TTL-based features.

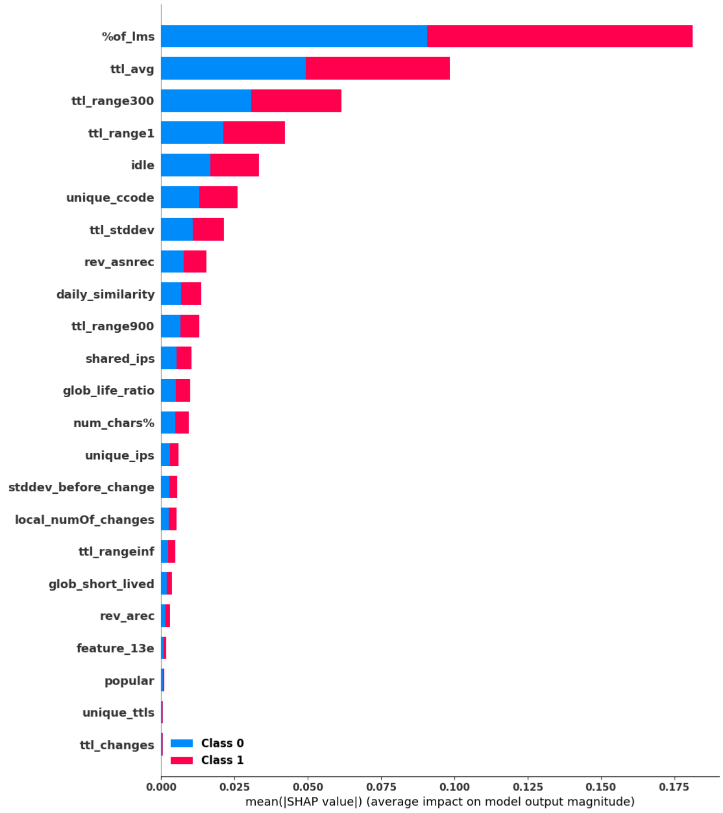


Figure 9: SHAP summary plot of feature contributions on KNN classifier.

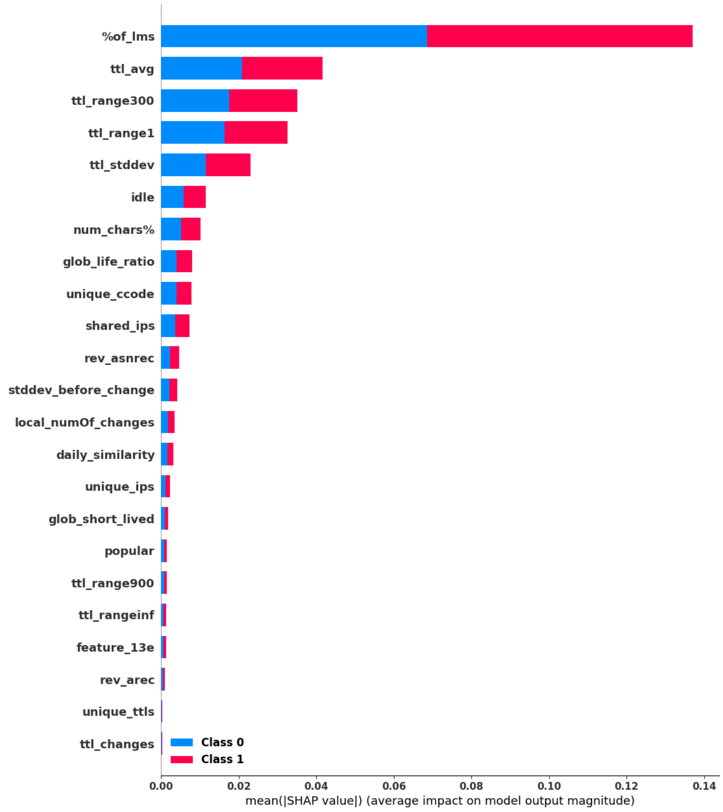
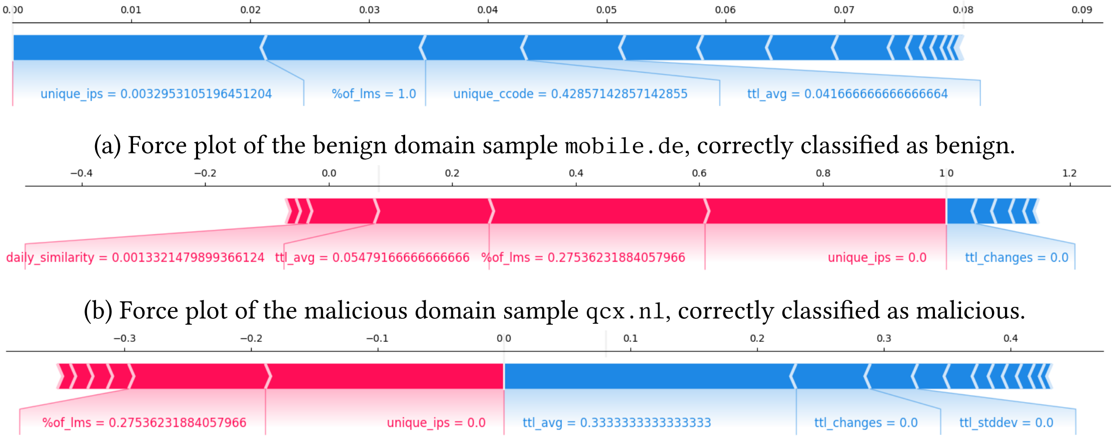


Figure 10: SHAP summary plot of feature contributions on SVC classifier.



(a) Force plot of the benign domain sample `mobile.de`, correctly classified as benign.

(b) Force plot of the malicious domain sample `qcx.nl`, correctly classified as malicious.

(c) Force plot of the malicious domain sample `szx.pw`, misclassified as benign.

Figure 11: Local explanations on other three domains from the dataset.