Danmarks Tekniske Universitet

DTU

Topic modelling for DNS misuse detection

Project report for 42186 Model-based machine learning

Authors

Lucas Brylle - s203832
Florian Comte - s231816
Tomas Durnek - s233788
Martin Jepsen - s124630

All members have contributed evenly to the creation of the project

September 19, 2024

Contents

1 Abstract

The Domain Name System (DNS) is a distributed service that provides a naming system for computers on the Internet [1, 2]. Misuse of the DNS protocol for malicious purposes is widespread, and can allow malicious actors to avoid detection [3, 4]. Detection of such misuse is therefore valuable for security monitoring. In this report, we consider how to perform unlabeled detection of DNS misuse, through discovery of DNS domain types. The type of a domain is considered to be a latent property, which represents the type of algorithm that generated the DNS domain. The type is assumed to influence the distribution and number of characters in the domain, allowing us to discover it using inference. We base our classification approach on ideas from topic modelling [5], and develop a model that can detect different types of DNS domains from unlabeled data.

Using the DGTA-BENCH dataset [6], we analyse domains coming from 2 different classes of misuse. The model achieves a classification accuracy of approximately 73.3% on the dataset, which is evaluated by using the labels. An empirical analysis of the learned types also shows that they align with the expected generation algorithms.

2 Background

2.1 DNS protocol misuse

Names in the DNS system are referred to as *domain names*, an example of which is `"inside.dtu.dk"`. Domain names are represented as strings consisting of *labels* separated by full-stop characters. Each label is made up of letters, digits, and hyphens. In the terminology of DNS, the example consists of 3 labels, which are `inside`, `dtu`, and `dk`. Each label in a domain can be at most 63 characters long, and the domain itself can be at most 255 characters long (including label separators), which is relevant for our modelling.

Detection of DNS protocol misuse on a network can serve as an indicator that malicious activity is occurring and is therefore valuable for security monitoring [3]. One common type of DNS protocol misuse is *DNS Tunneling* (DTA). This is a method of encapsulating other types of network traffic into the DNS protocol by encoding it as domain names [3]. An example of a DTA domain can be seen in Figure 1. Enterprises often permit DNS traffic through firewalls due to its critical functionality, inadvertently providing adversaries with an avenue to exfiltrate encoded data to servers under their control. Domain Tunneling Algorithms (DTA) thereby allow malicious actors to establish concealed channels for command and control (C&C) communications.

```
3d5d6f05f1dda3a096e8fa.2e10d53e935549b3a081982724c3e6f806.bad.ltd
```

Figure 1: Example of DTA domain containing hexadecimally encoded data.

Another type of misuse of the DNS protocol is Domain Generation Algorithms (DGAs). These are algorithms for randomly generating domains, which aim to make botnets harder to blacklist, and resistant to take-down efforts [4]. Examples of DGA domains can be seen in Figure 2. Malicious actors use DGAs by generating large amounts of domain names and

then using a small and changing subset of these for C&C. This forces defenders to spend large amounts of resources to mitigate attacks.

`qagbjxtcgmyccjmfb.com` `townshiphuman.com`

Figure 2: Examples of DGA domains.

2.2 Research question

Many existing approaches to DNS misuse detection focus on anomaly detection [7, 8] or supervised classification [9, 10, 11]. In this project, we instead choose an unsupervised approach. This has the advantage that the model is designed to identify previously unknown threats, for which there is no training data. The model can also be used for exploratory data analysis, which is useful for triaging DNS traffic on a network. Our project research question is:

Is it possible to create an unsupervised method for clustering and classification of DNS requests, with the goal of identifying DNS protocol misuse?

3 Dataset

3.1 Dataset overview and enrichment

To address the research question previously mentioned, we will use the DGTA-BENCH dataset [6]. This dataset is a substantial collection comprising 1.65 million domain names labeled by one of 55 different classes, based on the source (malware) from which they originate. To allow simplified analysis, we have enriched the dataset by further grouping the classes into *types*, which represent the domain generation algorithm type:

- **Benign** (1 class): Domains classified as benign (`legit`) in the original dataset.
- **DTA** (4 classes): Domains which are noted as being generated by DTAs in the description of the original dataset (`tuns`, `dnscapy`, `iodine`, and `dns2tcp`).
- **DGA** (50 classes): The rest of the domains.

Following this grouping, the distribution of the domains within the different types can be seen in Table 1.

Type	Count	Total %
Benign	1,080,696	65.2%
DGA	499,000	30.1%
DTA	77,537	4.7%

Table 1: Distribution of domain names.

As noted in the background information, different types of domain generation algorithms result in domain with different characteristics. We have decided to augment the dataset with 5 numerical features that can be seen in Table 2, which relate to the length and number of labels

in the domain. These features were selected based on expert input, which suggest that DGA domains are often short, and DTA domain are long. We will analyze these added features more in-depth in the Data Analysis subsection.

Feature	Description
domain_length	Length of domain name.
label_count	Number of labels in the domain.
min_label_length	Minimum length of a label in the domain.
max_label_length	Maximum length of a label in the domain.
avg_label_length	Average (mean) length of a label in the domain.

Table 2: Added domain features.

We have also chosen to implement tokenization of the domain names using Byte-Pair Encoding (BPE) [12]. The BPE algorithm splits the domains into tokens, which are units that may be larger than individual characters. In practice, tokenization finds the set of substrings, referred to as the vocabulary, which best represents the data based on frequency of occurrence.

A word in the vocabulary can range from an entire string, such as "google", to a single letter, such as "h". This approach allows us to maintain some of the high-level information within the encodings. For example the domain "inside.dtu.dk" could result in the tokens "inside", ".dtu", and ".dk", which would capture each label as an entire token. The purpose of this is to better represent the distribution of characters in the domains, since each token may model several consecutive characters. This could turn out to be beneficial, since malicious queries often contain encoded data within the DNS, that through tokenization will be more likely classified as malicious.

We set up our target vocabulary to 1024 tokens (words), including the special token with id 0 used for padding the domains. After the tokenization of each domain, they are padded to a common length using the padding token. This is done to allow vectorization of the code. In the conclusion, we discuss further tuning of the vocabulary size.

3.2 Data analysis

As explained previously, we have numerical and textual features. We will then divide the analysis into two parts.

3.2.1 Numerical features

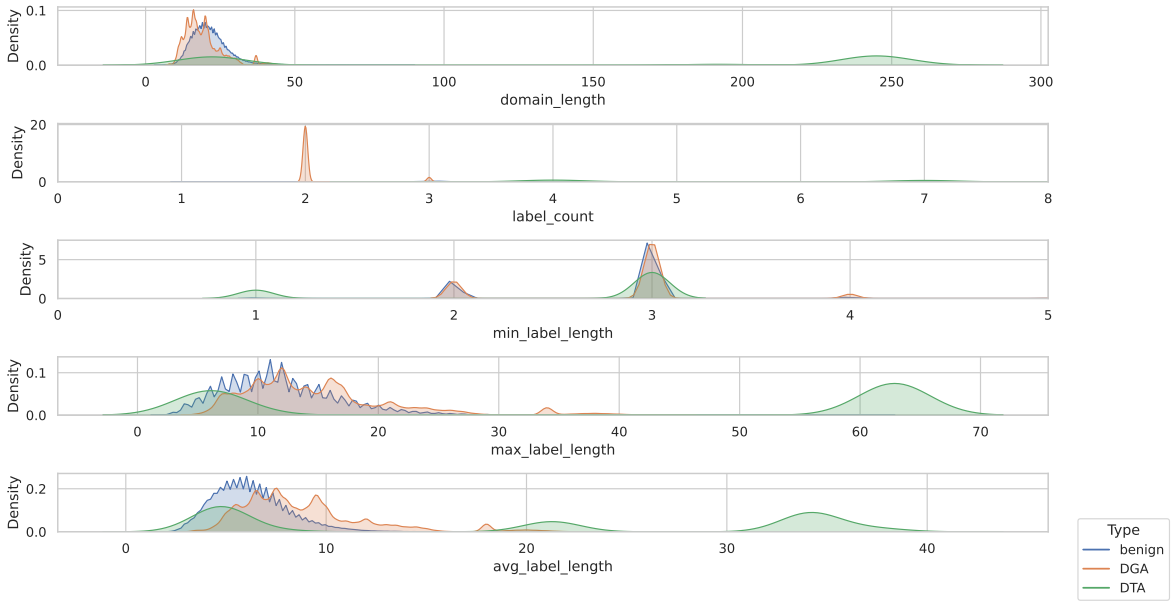
First, Table 3 summarizes the descriptive statistics for the numeric features extracted from the domain names in the dataset. These statistics include the minimum, maximum, mean, and standard deviation values for each feature.

Feature	Min.	Max.	Mean	Std. Dev.
domain_length	4	253	26.62	35.62
label_count	1	34	2.96	1.11
min_label_length	1	15	2.80	0.53
max_label_length	2	63	14.04	9.75
avg_label_length	1.15	39.5	7.63	4.69

Table 3: Statistics on numeric features

The wide range and high variability in the domain lengths and label lengths suggest the presence of diverse domain naming conventions, which may be indicative of different types of DNS traffic, including both benign and malicious activities. To further understand the differences in these numeric features based on the type of domain (DGA, DTA, benign), we visualize the data using density plots shown in Figure 3.

Figure 3: Density plots of numeric features



We directly see that the distributions are different among the types. The DTA domains can be extremely long and also short whereas benign and DGA domains are generally short (similar distribution with the benign one slightly shifted to the right). The encoding process inherent to DNS tunneling could be a reason for this observation as they may encode payloads within the subdomains, leading to unusually long domain names. Conversely, simpler tunneling techniques or minimal data encoding can produce shorter domains.

Benign domains typically have minimum label lengths of 2 or 3 characters, maximum label lengths of around 10-15 characters, and average label lengths of 3-8 characters. This reflects standard naming conventions where labels are concise and predictable. DGA domains exhibit similar patterns but with slightly bigger variability, as the randomly generated labels sometimes extend longer to ensure uniqueness. In contrast, DTA domains show a broader and

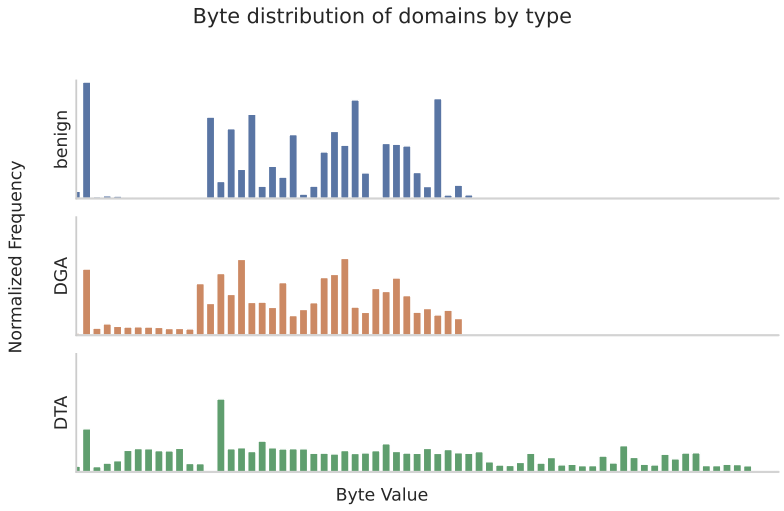
more varied distribution. This variability could be explained by the fact that our 'DTA' type includes different algorithms for domain generation (see the data notebook for more details).

Overall, the distinct distributions of the various numeric features confirm their suitability for use in our model. As noted in the last paragraph, DTA type (which contains data from 4 classes) is clearly seen to be divided, as the distributions seem to be normal mixtures.

3.2.2 Textual features

About the textual features, we can examine the domain's bytes distribution based on the type in Figure 4. We consider the byte distribution instead of token distribution since it is easier to visualize, and it still shows the expected behavior.

Figure 4: Domain's bytes distribution for each type



Benign domains: The byte distribution for benign domains is concentrated in a few specific areas, with several high peaks. This indicates that benign domains tend to have specific and consistent patterns in their byte composition. This is expected since benign domains usually follow standard linguistic and naming conventions, resulting in less variation in byte distribution.

DGA domains: The byte distribution for DGA domains looks more uniform than the benign one. This uniformity can be attributed to the random nature of DGA domain names. Unlike benign domains, which follow specific conventions, DGA algorithms generate a large number of random domains.

DTA domains: Finally the distribution of DTA domains is the most dispersed and uniform among the three types. This is consistent with the nature of DTAs, which may encode data within the domain names, resulting in a broader range of patterns and less predictability.

Overall, the differences observed in the byte distribution among benign, DGA, and DTA domains provide insights into their unique characteristics. This analysis supports the relevance of using tokens in our model. Given the disparities in byte distribution, token distribution is

expected to differ accordingly. Thus, integrating token-level features into our model is essential for capturing the nuanced characteristics of domain names and enhancing the effectiveness of classification.

4 Modelling

We have decided to base our approach on topic models [5]. These models are aimed at discovering latent topics within collections of documents. This is done by modelling each document as a finite mixture over an underlying set of topics. The topics are inferred by modelling them as a mixture of word probabilities, which are observed in the documents.

The fundamental idea behind our approach is that benign queries mostly contain human-readable domain names, while malicious queries often do not. The previously described DNS misuse algorithms either encode data into the query labels or randomly generate them. This will skew the label counts and character- or token-level distribution of the domain name, which was also observed in our data analysis.

Following the nomenclature of topic models, each DNS domain can be seen as a document chosen from one of K different topics. In the case of $K = 2$, this could represent the domain being either malicious or benign, and with a larger K, it can represent the individual algorithms that generated the domains. The topics determine the token distribution of the domains and the distribution of the numerical features derived from the labels.

A mixture model allows us to learn these topics from unlabeled data. This approach is unique because it can adjust to different types of DNS queries, even those created by previously unknown DGA and DTA methods, due to its flexibility and adaptability.

4.1 Topic model for DNS domains

The generative process for our model can be seen in Figure 5, and a PGM representation in Figure 6. Given N domains tokenized into T tokens each, the processed dataset consists of a matrix of tokens $\mathbf{B} \in \mathbb{N}^{N \times T}$, where each token $b_{n,t}$ is an integer between 0 and 1023. It also consists of length features $\{\boldsymbol{l}^{(n)}\}_{n \in N}$, where each domain has features $\boldsymbol{l}^{(n)} \in \mathbb{R}^5$.

Our model is designed with K topics, which can be seen as representing the classification of a domain. The distribution over the topics is given by weight vector $\boldsymbol{\pi}$, where π_k is the weight for topic k. The distribution over the tokens for each topic is given by the weight vectors $\{\boldsymbol{\theta}^{(k)}\}_{k \in K}$. Priors for the topic and token weights are chosen to be uniform. For a given domain of index n with topic k, the length features are assumed to be drawn from independent normal distributions as

$$\boldsymbol{l}^{(n)} \sim \text{Normal}(\boldsymbol{l}^{(n)} \mid \boldsymbol{\mu}^{(k)}, \boldsymbol{\sigma}^{2(k)})$$

This is a simplification chosen to deal both with count and average data. The implications of this choice are further discussed in the conclusion. The priors for the means $\boldsymbol{\mu}^{(k)}$ and variances $\boldsymbol{\sigma}^{2(k)}$ are chosen to be generic and uninformative [13, 14] (i.e. the means are i.i.d. standard normal).

Note that in step 2(c) of the generative process, the index t goes from 1 to T_n (the number

of tokens in domain n), to avoid modelling the padding token 0. In the model implementation, this is handled by masking to allow vectorization of the code.

1. For each topic $k \in \{1, \ldots, K\}$
   (a) Draw topic weight $\pi_k \sim \text{Gamma}(\pi_k | \frac{1}{K}, 1)$
   (b) Draw length feature means $\boldsymbol{\mu}^{(k)} \sim \text{Normal}(\boldsymbol{\mu}^{(k)} \,|\, \mathbf{0}, \mathbf{1})$
   (c) Draw length feature variances $\boldsymbol{\sigma}^{2(l)} \sim \text{HalfCauchy}(\boldsymbol{\sigma}^{2(l)} \,|\, \mathbf{1})$
   (d) Draw token distribution $\boldsymbol{\theta}^{(k)} \sim \text{Dirichlet}(\boldsymbol{\theta}^{(k)} \,|\, \frac{1}{1024} \cdot \mathbf{1})$
2. For each observed domain name $n \in \{1, \ldots, N\}$
   (a) Draw topic $z_n \sim \text{Categorical}(z_n \,|\, \boldsymbol{\pi})$
   (b) Draw length features $\boldsymbol{l}^{(n)} \sim \text{Normal}(\boldsymbol{l}^{(n)} \,|\, \boldsymbol{\mu}^{(z_n)}, \boldsymbol{\sigma}^{2(z_n)})$
   (c) For each query token $t \in \{1, \ldots, T_n\}$
      i. Draw token $b_{n,t} \sim \text{Categorical}(b_{n,t} \,|\, \boldsymbol{\theta}^{(z_n)})$

Figure 5: Generative process for DNS domain model.

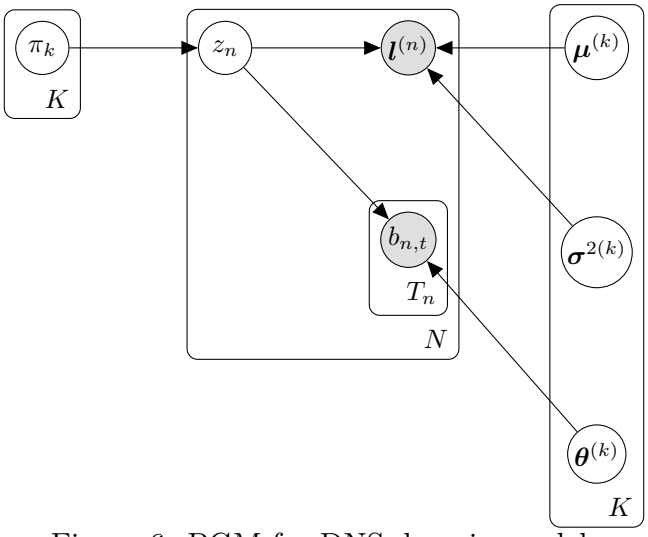


Figure 6: PGM for DNS domain model.

4.2 Training

The model was trained using Stochastic Variational Inference (SVI) [15]. We use the Adam optimizer [16] with a learning rate of $\alpha = 0.004$, decay rates $\beta_1 = 0.9, \beta_2 = 0.999$, and gradient clipping to a norm of 100. The training was run for 2000 steps, which was determined by empirically testing the accuracy of the model over a range of different choices. As discussed in the conclusion, the model has a tendency for posterior collapse if overtrained.

5 Results

To evaluate the performance of our model, we chose two scenarios: one with $K = 3$ (benign, DTA, and DGA) and another with $K = 55$ (the total number of algorithms in the DGTA-BENCH dataset). Our goal was to define well-separated clusters where we could achieve high accuracy and meaningful weights as expected from Table 1.

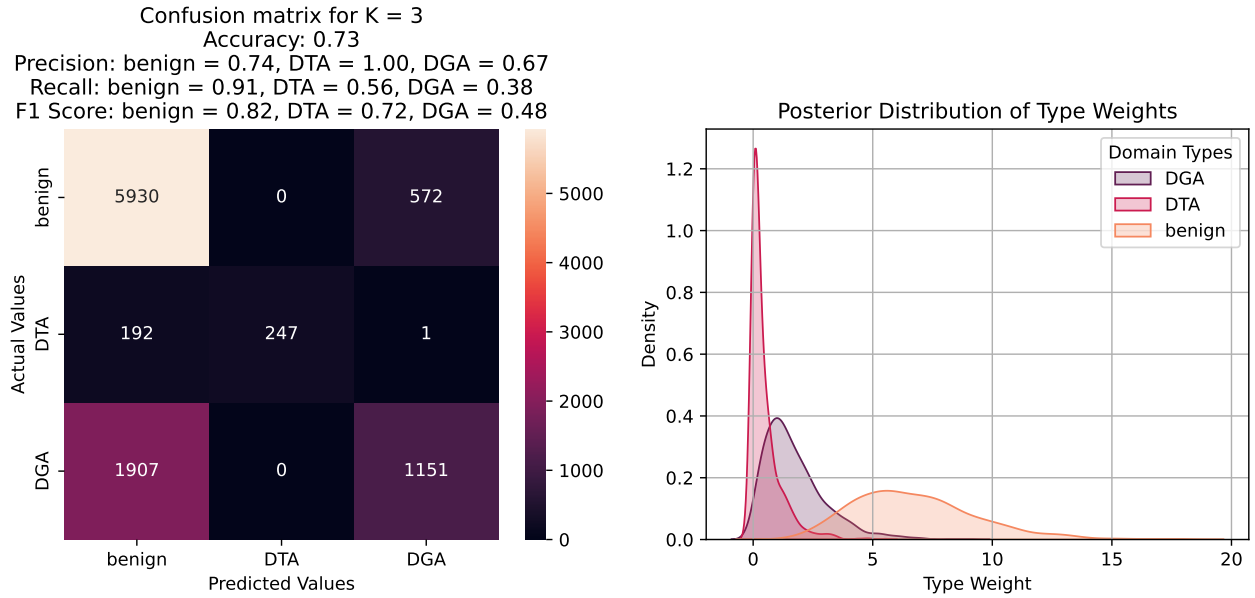


Figure 7: Confusion matrix and posterior distribution for $K = 3$.

We consider the accuracy of the labels assigned to individual domains, compared to the reference labels. Because the dataset is so large, we randomly sample 10000 domains to evaluate on. When determining the type of a domain we draw 1000 samples from the posterior distribution of z_n, and choose the most common value (the mode, or a MAP estimate). The accuracy of the inferred z_n is determined by mapping the known classes to the K different inferred types. To find the correct mapping we test each of the $K!$ permutations for small K, and select the mapping with the highest accuracy. For large K we instead use a greedy algorithm.

5.1 Model evaluation for K=3

For the $K = 3$ case, the confusion matrix and the posterior distribution of type weights can be seen in Figure 7. Our model, trained over 2000 steps with 10000 validation samples, achieved an accuracy of approx. 73.3%.

The precision metric, indicating the Positive Predictive Rate, shows a strong decision boundary for DTA with no false positives, reflected in a narrow density peak in the posterior. Precision for DGA and benign types is somewhat similar, but the recall (True Positive Rate) indicates a bias toward benign domains, resulting in a higher rate of true positives. The F1-score, the harmonic mean of precision and recall, summarizes these metrics, showing that while the model struggles with fully separating DGA domains, it excels at identifying DTA and benign domains. This performance might be influenced by the dataset's composition, where benign domains are more frequent than DGA and DTA.

5.2 Model evaluation for K=55

For $K = 55$, which corresponds to the total number of algorithms within the DGTA-BENCH dataset, our model achieved an accuracy of 53.8%. In comparison, a baseline model that always guesses benign would achieve an accuracy of 65.2% according to Table 1.

The model with $K = 55$ is more intricate and capable of detecting more complex patterns. This result demonstrates that the model can perform effectively in a setting that requires distinguishing between numerous algorithms. The lower accuracy relative to the baseline indicates the challenge of accurately classifying such a diverse set of algorithms. However, it also suggests that the model is not merely biased towards the most common class but is attempting to learn and classify the more subtle and complex distinctions between the 55 different types.

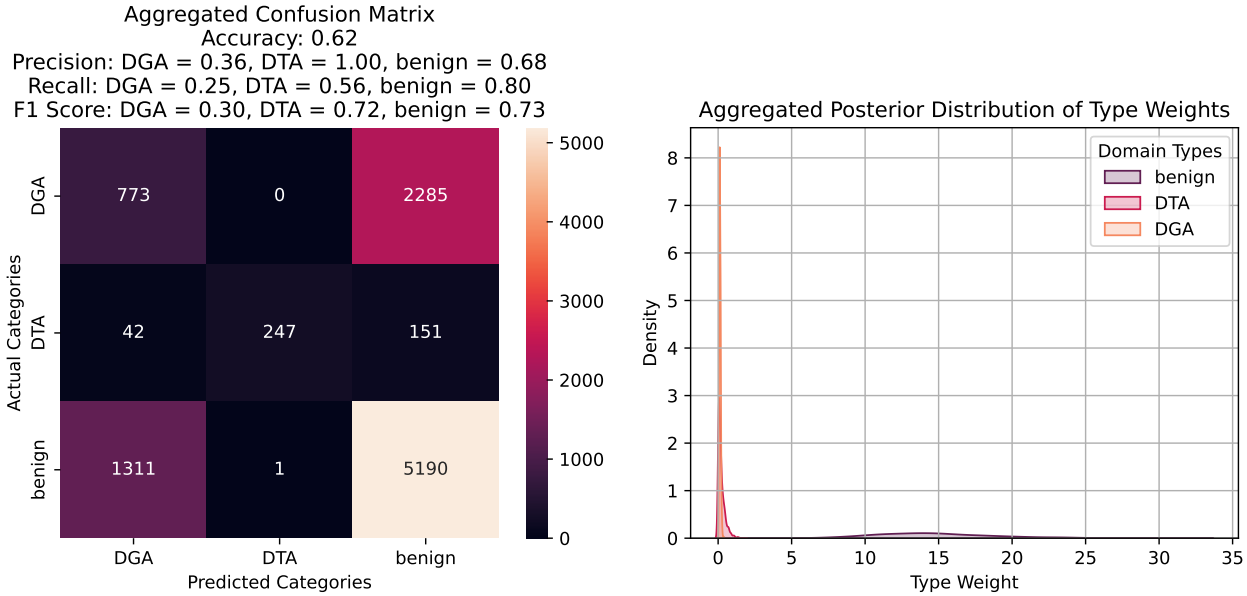


Figure 8: An aggregated confusion matrix for K=55, where all algorithms are grouped by the respective type

When clustering the domains into benign, DGA, and DTA types, the accuracy improved to 62.1%. The confusion matrix Figure Figure 8 shows that the model performs well in identifying benign domains but struggles with DGA and DTA. The posterior distribution indicates overlapping densities for DGA and DTA, with a greater density for DGA, reflecting its higher frequency in the dataset. Outside these fields, the benign domain dominates the weights.

5.3 Comparative analysis of K=3 and K=55

Comparing the results for K=3 and K=55, it is evident that the model can effectively identify benign domains in both cases. However, the performance drops when distinguishing between DGA and DTA, particularly in the K=55 scenario. This suggests that while the model scales well with the number of clusters, further refinement is needed for better separation of similar types.

6 Conclusion

We implemented an unsupervised model to detect latent topics within DNS domain names, discovering malicious activities DNS server traffic. To train our model, we made use of a dataset comprised of benign, as well as DGA- and DTA-generated domains. In addition, we

pre-processed the data by extracting 5 different numerical features. The analysis and plotting of these features suggested a significant, distinct variability in relation to the underlying classes.

Furthermore, we employed BPE tokenization to obtain the vocabulary used for training, maintaining some of the high-level label information. With extensive parameter tuning and experimentation, our model achieved an accuracy of around 73.3% when testing against the true labels. While we consider our results satisfactory, we also recognize that there are many opportunities for improvement with our model. At the same time, this section aims to clarify the approach to applying the model to unknown datasets and the techniques that would facilitate this.

6.1 Reconsideration of Prior Distributions

One of the most important areas of improvement that would enhance performance, as well as classification results, is choosing more appropriate priors for our data. Currently, we use Gaussian distribution for numerical values, which is fitting for averages but not suitable for label counts. A Poisson distribution would likely be a better fit for some of the counts while Gumbel distribution could be a good choice for minimum/maximum domain size modelling.

6.2 Vocabulary Size Tuning

While we experimented with various vocabulary sizes for tokenization, it remains interesting to tune this parameter, especially in relation to unknown datasets. A larger number of tokens will likely allow us to capture more high-level context within the domains, leading to better results. However, it may be computationally intensive and could potentially lead to overfitting if very infrequent tokens have a strong associated weight.

6.3 Optimal Topic Amount

For our dataset, we had access to both the domains and their corresponding labels (DGA, DTA, benign). Additionally, we were able to observe the underlying distribution by plotting the features based on these labels. This allowed us to make a good choice for the number of topics into which the model would classify the domains. Nevertheless, applying the model to a new, real-life scenario dataset would require making an educated guess without having access to the aforementioned metrics. One option would try different amounts of topics and decide which one yields the best results. Given that our model is Bayesian, we can compute the marginal posterior probability of the data across different options in order to determine the best K parameter.

6.4 Posterior Collapse

During the training of our model, we often experienced what is referred to as posterior collapse. This phenomenon occurs when the model disregards the relevant features and instead chooses different criteria to make predictions. In our case, most of the domains would be grouped into a single topic, resulting in an accuracy approaching the size of the largest group, which is benign with approximately 65% of the total samples. To address this issue, diverse regularization techniques could be adopted, for instance, the variational dropout [17].

References

[1] "Domain names - concepts and facilities." RFC 1034, Nov. 1987.

[2] "Domain names - implementation and specification." RFC 1035, Nov. 1987.

[3] Y. Wang, A. Zhou, S. Liao, R. Zheng, R. Hu, and L. Zhang, "A comprehensive survey on dns tunnel detection," *Computer Networks*, vol. 197, p. 108322, 2021.

[4] D. Plohmann, K. Yakdan, M. Klatt, J. Bader, and E. Gerhards-Padilla, "A comprehensive measurement study of domain generating malware," in *Proceedings of the 25th USENIX Conference on Security Symposium*, SEC'16, (USA), pp. 263–278, USENIX Association, 2016.

[5] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 05 2003.

[6] Y. Bubnov, "Dgta-bench - domain generation and tunneling algorithms for benchmark." Mendeley Data, 2021.

[7] J. Ahmed, H. H. Gharakheili, Q. Raza, C. Russell, and V. Sivaraman, "Real-time detection of dns exfiltration and tunneling from enterprise networks," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 649–653, IEEE, 2019.

[8] A. Nadler, A. Aminov, and A. Shabtai, "Detection of malicious and low throughput data exfiltration over the dns protocol," *Computers & Security*, vol. 80, pp. 36–53, 2019.

[9] S. Mahdavifar, A. H. Salem, P. Victor, M. Garzon, A. H. Razavi, and A. H. Lashkari, "Lightweight hybrid data exfiltration using dns based on machine learning," in *ICCNS '21: Proceedings of the 2021 11th International Conference on Communication and Network Security*, pp. 80–86, Association for Computing Machinery, 2021.

[10] T. Aurisch, P. C. ChacĂłn, and A. Jacke, "Mobile cyber defense agents for low throughput dns-based data exfiltration detection in military networks," in *2021 International Conference on Military Communication and Information Systems (ICMCIS)*, pp. 1–8, 2021.

[11] J. Steadman and S. Scott-Hayward, "Dnsxp: Enhancing data exfiltration protection through data plane programmability," *Computer Networks*, p. 108174, 2021.

[12] P. Gage, "A new algorithm for data compression," *C Users J.*, vol. 12, pp. 23–38, feb 1994.

[13] A. Gelman, D. Lee, and J. Guo, "Stan: A probabilistic programming language for bayesian inference and optimization," *Journal of Educational and Behavioral Statistics*, vol. 40, no. 5, pp. 530–543, 2015.

[14] Stan, "Prior choice recommendations." `https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations#generic-prior-for-anything`, 2024.

[15] M. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," 2013.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[17] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," 2015.