Detecting malicious DNS traffic with BERT

Netanel Indik[1], *Eden Dahary*[1], *David Hodefi*[1], *Gal Braymok*[1]

[1]School of Computer Science, Ariel University, Israel

netanel117@gmail.com, edendahary1@gmail.com, hodefi.david@gmail.com, galbraymok@gmail.com

Gal

Abstract

This research investigates DNS classification using BERT [1] embeddings and classifications to differentiate between benign and malicious DNS queries. The study utilizes a hybrid approach combining pattern analysis and semantic/syntactic analysis of domain names. The BERT model is trained on a dataset obtained from Akamai, consisting of DNS requests, responses, and verdicts from a DNS security system.The research contributes to the field of DNS security by advancing DNS classification models using a solutions from the NLP world.

Eden

1. Introduction

Malicious Domain Name System(DNS) detection is a rapidly evolving field that is concerned with identifying a malicious or non-malicious domain based on their pattern. It has become an increasingly important area of study in recent years. The main objective of such research is to develop algorithms and techniques that can accurately and reliably identify on what domain is malicious or benign. This involves analyzing the query of the domain including the name of the domain, response time, its type and the answer we got. Domains can be broadly classified into four categories: BENIGN, MALWARE, PHISHING, Command and Control (CNC). All this categories related to DNS.

DNS is a fundamental protocol used on the Internet to translate domain names into IP addresses. It provides a hierarchical naming system that allows users to access websites and other network resources using human-readable domain names rather than numeric Internet Protocol (IP) addresses. However, DNS is also a prime target for malicious activities. Attackers can use DNS to evade detection, exfiltrate data, and control infected devices, among other malicious purposes. One technique that attackers use is Domain Generation Algorithms (DGA), which generates a large number of domain names that can be used to communicate with a CNC server. The domain names are typically generated based on a predefined algorithm that takes inputs such as the current date and time, making them difficult to predict or detect. The attacker can register one of these domain names and use it to control infected devices or exfiltrate data. Detecting DGA is challenging because the domain names are generated on the fly, and the actual CNC server is not known beforehand.

To detect and prevent such malicious activities, researchers have proposed various techniques, including signature-based detection, anomaly-based detection, and machine learning-based detection. There a lots of techniques to identify malicious DNS, we will speak on two of them: DNS Sinkholing and Machine Learning Based Analysis.

DNS Sinkholing involves redirecting malicious DNS traffic to a controlled server or IP address. This technique is used to intercept and block connections to malicious domains or IP addresses. By redirecting the traffic to a sinkhole server, organizations can analyze the DNS queries and responses, identify malicious patterns, and take appropriate actions to mitigate the threat. Sinkholing can be an effective technique for detecting and preventing malware infections, botnet communications, and other malicious activities relying on DNS.

Machine Learning-Based Analysis: Machine learning techniques are increasingly being used to detect malicious DNS activities. By training machine learning models on large datasets of DNS queries and associated metadata, patterns and anomalies can be identified to distinguish between benign and malicious behavior. These models can learn to recognize known patterns of malicious DNS activities, such as Domain Generation Algorithm (DGA), fast-flux networks, or suspicious DNS query patterns. Machine learning-based analysis allows for real-time monitoring and detection of emerging threats and can help in identifying previously unknown attacks.

In this work, we propose a novel DNS detection approach based on deep learning techniques, which target the malicious DNS. The input for our approach is a stream of DNS traffic logs that are constantly grouped by domain. The logs are collected over a sufficiently long period of time to allow the detection of malware. We will use Bidirectional Encoder Representations from Transformers (BERT), which is a Natural Language Processing (NLP) model developed by Google. It is based on the Transformer architecture, which is a type of neural network designed to process sequential data efficiently. BERT is specifically designed for pre-training language representations, which means it can be trained on a large corpus of unlabeled text to learn contextualized word embeddings. The model captures the contextual information of each word by considering the surrounding words on both sides and generates high-quality word embeddings that capture the meaning of the word in its context. We will use BERT in order to classify if the domain is malicious or benign, we take from our input the the domain name and its answer which provide from AKAMAI [2], We decided to take the answer from the query because it contains essential things, in order to use it on BERT we take the answer and covert it to a sentence and then we will use classification BERT with fine tuning on this sentence. The idea behind this is that we can detect attacks by finding anomalies in the patterns of the structure and content of the answer.

Identify malicious DNS is crucial for companies to protect their networks and systems from various cyber threats. Malicious DNS detection helps in preventing potential security breaches and attacks. By identifying and blocking access to malicious domains, companies can proactively mitigate the risks associated with malware infections, phishing attempts, and other malicious activities. This proactive approach significantly reduces the likelihood of successful attacks and helps maintain

a secure environment.

Another advantage is that Many industries and sectors have specific compliance and regulatory requirements related to cyber security. Implementing robust malicious DNS detection mechanisms helps organizations meet these requirements and demonstrate their commitment to safeguarding data and protecting sensitive information. Compliance with regulations not only mitigates legal and financial risks but also enhances the organization's reputation and customer trust.

However Detecting malicious DNS can pose several challenges due to the ever-evolving nature of cyber threats. One major problem is that malicious domains creators are developing new techniques to bypass detection systems. They may use tactics such as DGA, fast-flux networks, or domain cloaking to make their malicious domains appear legitimate or constantly change their domain names to evade detection. Which makes it hard to detect the new malicious domains.

Another problem is the large Volume of DNS traffic that organizations generate, they are creating significant amount of DNS traffic on a daily basis, making it challenging to analyze and distinguish between benign and malicious DNS activities. Detecting anomalies and patterns associated with malicious DNS within this large volume of data requires advanced analysis techniques and scalable infrastructure.

The rest of the paper is structured as follows: Section 2 describes the work that has been done in this field and a literature review of their research. Where paying attention to their architecture and general approach and the results of their work. Section 3 describes our dataset, from where we get the dataset, what fields we have and how much benign and malicious. Section 4 describes our model with what tools we create our model and all its architecture Section 5 show our results for the proposed model across our dataset. Finally, Section **??** **??** concludes and summarizes this paper. For ease of reading, we provide list of abbreviations in Table 1

Table 1: *List of Abbreviations*

Abbreviation	**Meaning**
DNS	Domain Name System
IP	Internet Protocol
NLP	Natural Language Processing
DGA	Domain Generation Algorithm
CNC	Command and Control
BERT	Bidirectional Encoder Representations from Transformers
TCN	Temporal Convolutional Network
LSTM	Long Short-term Memory Network
CNN	Convolutional Neural Network
TLD	Top Level Domain
URL	Uniform Resource Locator
HTML	HyperText Markup Language
Whois	A widely used Internet record listing that identifies who owns a domain and how to get in contact with them
NB	Naïve Bayes
RF	Random Forest
CDN	Content Delivery Network
FPR	False Positive Rate
RR	resource record

Eden

2. Related Work

In this section we provide a brief review of related prior work. It is important to note that all of them use domain classification but, none of them use BERT on their research. There have been lots of research on this subject that show how to detect a malicious URL domain.

In [3] research they combine three models and used them together to detect whether the URL is malicious or non-malicious in a improved way, they want to achieve better results than the previous models that they saw like the combined model of Convolutional Neural Network (CNN) and and Long Short-term Memory Network (LSTM) or the combined model of CNN and Temporal Convolutional Network (TCN). First, they took a dataset of DGA and take only the Top Level Domain (TLD) name when all the data contain 54 different types of DGA and separate labels for each type of DGA they were practice on 100K of malicious and 100K of non-malicious that were taken for bunch of sources. The models its-self contain three layers : input layer - encode and convert the domain name. feature extraction layer - the feature extraction layer is the basic body of the whole malicious domain name detection model. It is divided into three convolution layers according to different convolution core sizes, namely, the convolution network with convolution core sizes of 3, 4, and 5. Each convolution network has two identical convolution layers, and each convolution layer includes a convolution layer, standardization layer, activation layer, and pooling layer and a time convolution network (TCN) is added behind each convolution network. Then input the feature extraction results of each convolutional. result output layer - The result output layer mainly contains a layer of LSTM with attention mechanism and two layers of fully connected layers with different numbers of neurons. The results were significant which provide Precision of 99.12%, Recall 99.76%, Accuracy 98.81%, False Positive Rate(FPR) 1.52% and F1-score 0.8935%, while the other models provide results that are less than 94%. The malicious domain name detection model optimizes the model and improves the detection accuracy by adding TCN network structure and LSTM network with attention mechanism on the basis of CNN network structure. The malicious domain name detection model mainly identifies the nature of malicious domain names by extracting the character-level features of domain name datasets. The advantage of their work was that they doesn't need to manually extract features like other detection models based on a deep learning algorithm. However, there were still some problems in the their model, such as long training time and strong dependence on the size of the dataset.

In [4] research they took their main goal to create a perfect feature engineering, they extract useful features from the dataset and selecting the most optimal features using different features selection techniques to build and compare the accuracy of a set of intelligent models. They want to take the most essential features for malicious URL classification and to identify which features are the most effective in carrying out accurate predictions on identifying malicious URLs. The most used features can be classified into three main classes: 1. Lexical-based - collected through lexical scanning and are obtained from the URL name/URL string itself. 2. Content-based - reflect the web page content, including its HTML tags, iframes, lines, hyperlinks, JavaScript functions, and keywords that are mostly extracted by Computational Intelligence and Neuroscience. crawling the web page using tools such as Selenium WebDriver 3. Network-based - combine Network, DNS, and host-based features and

include latency, DNS query data, domain registry data, payload size, WHOIS queries information about domain names, and Ips. The dataset that they used was the Malicious and Benign webpages dataset, a publicly available dataset released by Singh and Kumar in 2020. The data were collected by crawling the Internet using the Mal Crawler tool, and the labels were verified (as good or bad) using Google Safe Browsing (API). The dataset contained various attributes from websites, namely, URL, IP address, JavaScript code, obfuscated code, geographical location, top-level domain, and HTTPS, all of which can be useful for classifying a web page as either malicious or benign. The dataset also included raw page content including the JavaScript code that can be used to extract further attributes. The dataset contained two sets: one set of webpages represents the train data which comprised 1.2 million records and 11 attributes and the other set of webpages represent the test data, which comprised 0.364 million records and 11 attributes. In the train dataset, 27,253 were labelled malicious and 1,172,747 were labelled benign. In the test dataset, 353,872 URLs were labelled benign and 8,062 were labelled malicious. In this research Naïve Bayes (NB) achieved slightly higher values than the other models, achieving accuracy of 96.01%. The highest achieved precision of 95.64% was by NB, which was followed by Random Forest (RF), CNN, and then LSTM, which obtained precision between 87.24% and 87.31%. However, the recall achieved by NB was the lowest as all the other models achieved a recall of 100%. The advantage of their work was that they didn't use a huge model architecture instead they used and focus only on features engineering and with a normal model they achieve good result. However, if they were using a more complex model architecture they could predict better result.

In [5] research they proposed a method for detecting both tunneling and low throughput data exfiltration over the DNS. They focus on detecting and denying requests to domains as an effective data leakage shutdown. Therefore, their proposed solution handles streaming DNS traffic in order to detect and automatically deny requests to domains that are used for data exchange They extract features based on the querying behavior of each domain, and after an anomaly detection model is used to classify domains based on their use for data exfiltration. Their method was evaluated on a large-scale recursive DNS server's logs with a peaking high of 47 million requests per hour. Within these DNS logs, they injected data exfiltration traffic from DNS tunneling tools. They extract 6 features from the DNS query: Character Entropy- they compute the entropy on per domain queries to detect encrypted or encoded querying behavior. type ratio : corresponds to the non-IP type ratio -The feature that they create computes the rate of A(IPv4) and AAAA (IPv6) records for per domain. qname val : Unique Query Volume - DNS traffic is rather sparse as responses are largely cached within the stub resolver. However, in the case of data exchange over the DNS, the domain-specific traffic is expected to avoid cache by non-repeating messages, or short time-to-live in order for the data to make it to the attacker's server. Avoiding cache as well a lengthy data exchange, might result in a higher volume of requests compared to a normal setting. qname len: Query Length Average - As a complementary to the volume feature and given a query size limitation, there is a trade-off between the volume of queries and their length, hence making it an effective feature for detection. unique query ratio : corresponds to the unique query ratio - A domain whose subdomains are used as messages is not likely to repeat them. Therefore, when comparing domains used for exfiltration to normal domains we expect to see a much higher unique query ratio for the latter. avg longest word len - For each primary domain, each subdomain is decomposed to its hierarchy labels ordered by their length. Starting from the longest to the shortest substring, an English dictionary word lookup is performed over the current substring. If the lookup succeeds the length of the substring is taken as the longest meaningful word length. This length is then divided by the length of the subdomain and averaged out over all of the subdomains. While English is not the only language used on the Internet, this may help distinguish domains with a large number of domains to either readable, or non-readable. The main dataset that they used was a single week of DNS traffic collected from a subset of recursive DNS resolvers operated by Akamai Technologies. Dataset is considered a large-scale DNS traffic sample with at least a hundred thousand end users, an average of 35 million hourly requests with a standard deviation of 74 million hourly requests. They used two real-life malware : Framework-POS - previously used for the theft of 56M credit cards from Home Depot in 2014, and Backdoor.Win32.Denis, which was active in the Cobalt Kitty APT in 2016. They used all their work based on real dataset with a large-scale DNS traffic and they get good result, they also extract features from them. Although they extract features they didn't use BERT on the qname which could lead to a better result.

Finally, we mention a proposed solution based on a new standard which detect difference between legitimate and malicious DNS by using the BERT. BERT is a pre-trained language model that has revolutionized the field of NLP. The huge advantage of BERT is its ability to capture contextual nuances and dependencies in language, allowing it to understand the meaning of words within the context of a sentence.

3. Dataset

Netanel

The origin of the dataset used in this research is Akamai company. The data was collected by one of Akamai's products; a DNS protection system. The data was obtained passively by observing the communications between end hosts and their DNS resolvers. The dataset contains info about DNS requests and their responses as well as the verdict of the DNS security system; whether the request is malicious or benign. The system relies on blacklists to determine that.

The dataset consists of the following fields:

Table 2: *DNS Traffic Dataset Fields*

Field	**Description**
request_ts	Time of the DNS request
response_ts	Time of the DNS response
client_token_dec	Decoded client (company) ID
qname	Domain name (URL) for the IP request
qtype	Type of DNS record being queried
answer	Response packet from the DNS server
nx_domain	Indicates if the domain name couldn't be resolved
l1_cache_hit	Indicates if the resolution was found in the cache
response_size	Overall size of the DNS response
category_id	Category of the domain (e.g., CNC, MALWARE, PHISHING)
list_id	Blacklist from which the category is identified
is_malicious	Main label indicating if the domain is malicious or benign

Possible limitations of Host-Resolver collected data would be: It describes only the behavior of a specific group of organizations. Hence, general patterns relating to malicious behavior may not be detected by this type of analysis. [6] Furthermore, this data was captured over a short timeframe, a single hour. activity of botnets for example could be missed if the bots create timed requests at different times of the day. The data consists of 16.7M benign and 16K malicious DNS requests.

As can be seen below, the malicious attacks in this dataset are malware, phishing, and CNC.

Attack Type	Count
Benign	16,756,744
Partially Malicious (Malware)	12,369
Partially Malicious (Phishing)	2,567
Malware	961
CNC	66
Phishing	10

Table 3: *Distribution of Domain Types*

4. Framework

Netanel & Eden

We offer an hybrid approach that utilizes both finding patterns of malicious domains in the answer part of the response as well as semantic and syntactic analysis of the domain name to intercept domain names generated by DGA. We will use BertForSequenceClassification with the pretrained "bert-base-uncased" model for the task. BertForSequenceClassification is a Bert Model transformer with a sequence classification/regression head on top (a linear layer on top of the pooled output).

4.1. Data preparation

The dataset contains DNS traffic of requests that got answered by the cache level and requests that resulted in nx_domain, meaning that any resolution could not be provided by the DNS server. While even these kind of requests could be malicious, their answer part would be empty and therefor our approach wont catch them or in the case of the cache hit, it could be catched in the first query before the domain name got cached. Hence, we removed all requests with empty answers.

4.2. Modeling the structure of the answer as a sentence for BERT

The answer part of a DNS response contains the actual resource records that provide the requested information.

Each entry in the answer part typically represents a resource record (RR) that holds specific data related to the query. The type of resource record determines the type of information provided in the answer. Here are some common types of resource records:

Table 4: *DNS Record Types*

Record Type	Description
A	IPv4 address.
AAAA	IPv6 Address
NS (Name Server)	authoritative name servers
CNAME (Canonical Name)	alias for a domain name.

The answer part may contain multiple resource records, depending on the query and the DNS server's response. Each resource record typically includes the domain name, the record type, the record data (such as IP addresses, domain names, etc.), and additional information like the time to live (TTL) value, which indicates how long the information can be cached before it should be refreshed.

Analyzing the answer can help us detect malicious activity [6]. Anomalies in the structure of the answer, for example the length of the answer, the TTLs, the number of NS records, the use of CNAMEs and the number of IP records can indicate malicious activity. The next table shows a comparison between malicious and benign answers.

Table 5: *Answer stats comparison*

Metric	Benign	Malicious
response_size mean	176.713	134.591
duration mean	73.414	150.455
mean_ttl mean	6862.931	4224.926
cnt_ans_parts mean	3.633	3.627
cnt_ans_A mean	1.775	2.945
cnt_ans_AAAA mean	0.090	0.045
cnt_ans_NS mean	0.007	0.000
cnt_ans_CNAME mean	1.748	0.618

The comparison reveals distinct differences between malicious and benign answers. We observe that benign traffic has a significantly higher mean response size and longer mean TTL, while malicious traffic tends to have a greater number of IPv4 records and a higher occurrence of CNAMEs.

To leverage the capabilities of BERT in solving our problem, we model the answer as a sentence that represents the domain name and the structure of the answer. For example, if we have the domain name:

`ex0.example0.com`

Table 6: *Answer example*

Class	Name	TTL	Type	Value_str
IN	ex.example.com	27940	5	ex1.example1.com
IN	ex.example.com	500	5	ex2.example2.com

And the corresponding answer, the formatted answer sentence would be:

`ex0.example0.com URL 27940 CNAME URL URL 1596 CNAME URL`

In this example, the sentence starts with the qname(the queried domain name) and followed by the answer parts concatenated one after another. Each domain name replaced with "URL" and each IP address replaced with "IPv4" or "IPv6".

The numerical part replaced with its corresponding record type. e.g. CNAME, NS, etc.

To ensure uniqueness, we keep only the unique formatted answers by dropping duplicates. Answers differ only by IP address considered duplicates for this matter. At this point our data consists of formatted answers and their corresponding classification as malicious or bening. Each row represent a unique answer structure.

4.3. Training BertForSequenceClassification

To train BertForSequenceClassification we initially needed to use BERT tokenizer. It tokenizes the sentence into subword tokens. BERT uses a WordPiece tokenizer, which splits words into smaller subwords or character-level tokens. This step helps to handle out-of-vocabulary words and improve generalization. In our case the tokenization will provide a different view of the domain name as it splits out-of-vocabulary words into smaller meaningful chunks.

The next step involves obtaining the BERT embeddings from the tokenized formatted answer. Finally, we fine-tune the model specifically for this DNS classification problem by adding a classification layer on top of the embedding.

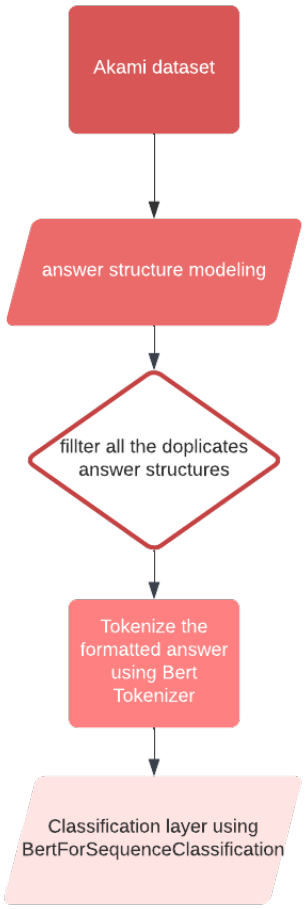

Figure 1: *model architecture*

David

5. Experimental Evaluation & Results

This section presents the results for the proposed model across our data set. In the experiment, the following training parameters were used: batch size of 16, evaluated and optimized every epoch step, aver all we trained 3 epochs.

Detecting malicious DNS queries is often more important than detecting benign DNS queries due to the potential consequences associated with malicious behavior. In many scenarios, the impact of a false negative (misclassifying a malicious DNS queries as benign) can be significantly more severe than a false positive (misclassifying a benign DNS queries as malicious). Malicious DNS queries can pose security threats, cause financial losses, compromise user privacy, or lead to other harmful outcomes. Therefore, it is crucial to prioritize the identification of malicious DNS queries to ease risks and protect systems and users.

Additionally, our data set is imbalanced, with a larger number of benign DNS queries compared to malicious DNS queries, relying solely on accuracy can be misleading. Accuracy alone does not consider the class distribution and may be biased towards the majority class. In imbalanced data sets, a classifier that labels all DNS queries as benign can still achieve high accuracy if the majority of DNS queries are indeed benign. This can result in poor performance when it comes to detecting the minority class, which in this case is the malicious DNS queries. Therefore, it is necessary to consider evaluation metrics such as precision, recall, and the F1 score that take into account the true positive rate and false positive rate for each class, providing a more comprehensive understanding of the model's performance, particularly for the minority class.

Therefore we evaluate our result in a few methods based on the result that we got from the classification model, accuracy, precision, recall, specificity and the F1 score.

Table 7: *Overall Performance*

Field	**Value**
Accuracy	99.946%
Precision	99.962%
Recall	99.983%
F1 Score	99.972%

We also want to show the macro averages corresponding to our result:

Table 8: *Macro averages*

Field	**Value**
Malicious DNS Queries	
Precision	77.5%
Recall	88.571%
F1 Score	82.666%
Macro-average	
Precision	88.731%
Recall	94.277%
F1 Score	94.2715%

The results above of this study presents an evaluation of our model of classification,

From the obtained results using the BERT classification model on the formatted answer as shown in paragraph Frame-

Table 9: *Confusion Matrix*

True Class / Predicted Class	Malicious	Benign
Malicious	24116	9
Benign	4	31

work, we can draw several conclusions. Firstly, the model demonstrates a high level of accuracy in predicting benign instances, with 24116 correct classifications out of 24125 instances. This suggests that the model is effective in identifying non-malicious cases accurately.

However, the model's performance in detecting malicious instances shows room for improvement. Out of 35 actual malicious instances, only 31 were correctly identified, indicating that the model missed a few instances. This highlights the importance of further refining the model's ability to recognize and classify malicious instances with higher precision and recall.

Additionally, given the significant class imbalance in the test dataset, with a much larger number of benign DNS queries compared to malicious DNS queries, it is crucial to consider evaluation metrics beyond accuracy. Metrics such as precision, recall, and F1 score provide a more comprehensive assessment of the model's performance, especially for the minority class. By analyzing these metrics, we can gain a deeper understanding of the model's strengths and weaknesses in differentiating between benign and malicious DNS queries.

In conclusion, while the BERT classification model demonstrates proficiency in identifying benign DNS queries, further efforts should be directed towards enhancing its capability to detect and classify malicious DNS queries accurately. This could involve fine-tuning the model, augmenting the training data, or exploring other techniques to address the class imbalance issue. By continuously refining the model's performance, we can enhance its effectiveness in real-world applications where the detection of malicious DNS queries is of utmost importance. We furthermore discuss ways for improvement in the next paragraph.

Gal & David

6. Discussion, Conclusions, Future Work

The experimental evaluation of our DNS classification models yielded promising results, although our performance did not reach the accuracy of prior research[5]. Giving our problem space - a real world 'Host-Resolver' dataset that consists of an imbalanced class, however by using the BERT classification model, which utilizes BERT embeddings as input, we demonstrated a significant improvement in the field because it is robust and can be implemented in a variety of languages.

For future work, we could fine-tune the model to improve the BERT classification model's performance in detecting and classifying malicious DNS queries [**?**]. Fine-tuning also allows for the optimization of hyperparameters and architectural choices specific to the task of detecting malicious DNS queries [7]. This involves training the model with additional data that specifically focuses on various types of malicious activities. By including diverse examples of attacks and variations in malicious behavior, the model can learn more nuanced representations and improve its overall accuracy.

In addition, augmenting the training data is essential for enhancing the BERT model's performance. Increasing the size and diversity of the training data can be achieved by collecting data from multiple sources, organizations, and time periods [8]. This will provide a broader range of malicious activities and help the model generalize better. Augmentation techniques, such as data synthesis or oversampling of the minority class (malicious queries) [9], can help address the class imbalance issue and provide a more balanced training set. Additionally, incorporating data from different geographical regions and industry sectors can improve the model's ability to detect and classify malicious queries in various contexts [10].

Moreover, we should explore more advanced techniques. While BERT has shown effectiveness, exploring other advanced techniques in NLP and machine learning can further enhance the model's performance. Investigating transformer-based models like GPT or XLNet [11], which have demonstrated strong performance in various NLP tasks, could be beneficial. These models may capture more complex patterns and relationships in DNS queries, leading to improved accuracy in detecting malicious instances.

In order to further improve the accuracy and robustness of our DNS classification models, it is imperative to consider the integration of both NLP models [12], such as BERT, and ML models that leverage statistical analysis of DNS queries.

Combining NLP models like BERT with ML models can offer complementary advantages. While BERT excels in capturing semantic relationships and contextual information, ML models can provide [13] valuable insights by analyzing statistical patterns within DNS queries. By leveraging ML techniques, such as feature extraction and statistical analysis, we can uncover hidden patterns and correlations in the data that may enhance the detection and classification of malicious queries.

Integrating ML models into our approach would involve designing algorithms that can extract relevant statistical features from DNS queries [14]. These features could include the frequency distribution of query types, temporal patterns, or statistical measures like mean and standard deviation. By incorporating such statistical analysis, our models can gain a deeper understanding of the underlying patterns and anomalies associated with malicious DNS queries.

Additionally, we can explore ensemble learning techniques to combine the strengths of different models. Ensemble learning [15] involves training multiple models and combining their predictions to make a final decision. By combining both NLP models like BERT and ML models that leverage statistical analysis, we can create a more robust and accurate classification system. Ensemble learning can help mitigate biases and improve overall performance by aggregating the predictions from diverse models.

We think that it is necessary to address the limitations of the current dataset and analysis approach. In future studies, efforts should be made to acquire data from a more diverse set of sources. This will help overcome the limitation of capturing only the behavior of a specific group of organizations and enable the detection of more general patterns of malicious behavior. Extending the data collection period beyond a single hour is also important to capture the activities of botnets or timed attacks that may occur at different times of the day. Moreover, integrating other sources of information, such as reputation lists, threat intelligence feeds, or anomaly detection algorithms, can complement the DNS data and enhance the model's ability to detect and classify malicious queries accurately.

In summary, our research has laid a solid foundation for DNS classification using BERT embeddings and classifications [16]. By addressing the outlined future work and limitations, we can improve the accuracy, robustness, and overall effectiveness

of our DNS classification models, thereby advancing the field of DNS security.

Gal

7. References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] "Akamai technologies," https://www.akamai.com/, accessed: June 11, 2023.

[3] X. Huang, H. Li, J. Liu, F. Liu, J. Wang, B. Xie, B. Chen, Q. Zhang, and T. Xue, "A malicious domain detection model based on improved deep learning," *Computational Intelligence and Neuroscience*, vol. 2022, pp. Article ID 9 241 670, 13, 2022.

[4] A. Alhogail and I. Al-Turaiki, "Improved detection of malicious domain names using gradient boosted machines and feature engineering," *Information Technology and Control*, vol. 51, no. 2, pp. 313–331, 2022.

[5] A. S. Asaf Nadler, Avi Aminovb, "Detection of malicious and low throughput data exfiltration over the dns protocol," *Computers & Security.*, vol. 80, no. 21, pp. 36–53, 2019.

[6] Y. Zhauniarovich, I. Khalil, P. S. Yu, and M. Dacier, "A survey on malicious domains detection through dns data analysis," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 67:1–67:36, 2018.

[7] K. Ameri, M. Hempel, H. Sharif, J. Lopez Jr., and K. Perumalla, "Cybert: Cybersecurity claim classification by fine-tuning the bert language model," *Journal of Cybersecurity and Privacy*, vol. 1, no. 4, pp. 615–637, 2021. [Online]. Available: https://www.mdpi.com/2624-800X/1/4/31

[8] S. Mahdavifar, N. Maleki, A. H. Lashkari, M. Broda, and A. H. Razavi, "Classifying malicious domains using dns traffic analysis," in *2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 2021, pp. 60–67.

[9] S. Shafieian, D. Smith, and M. Zulkernine, "Detecting dns tunneling using ensemble learning," in *Network and System Security*, Z. Yan, R. Molva, W. Mazurczyk, and R. Kantola, Eds. Cham: Springer International Publishing, 2017, pp. 112–127.

[10] W. Yang, Y. Xie, L. Tan, K. Xiong, M. Li, and J. Lin, "Data augmentation for bert fine-tuning in open-domain question answering," 2019.

[11] M. O. Topal, A. Bas, and I. van Heerden, "Exploring transformers in natural language generation: Gpt, bert, and xlnet," 2021.

[12] H. Face, "Bert 101," https://huggingface.co/blog/bert-101, 2022.

[13] Databricks, "Automating ml scoring and alerting for detecting criminals and nation states through dns analytics," https://www.databricks.com/blog/2022/08/02/automating-ml-scoring-and-alerting-for-detecting-criminals-and-nation-states-through-dns-analytics.html, 2022.

[14] Hindawi, "Performance evaluation of deep learning architectures for network intrusion detection systems," https://www.hindawi.com/journals/jcnc/2021/4767388/, 2021.

[15] Springer, "A time-based approach for detecting dga domains using deep learning," https://link.springer.com/chapter/10.1007/978-3-030-36938-5_32, 2019.

[16] G. Lee, M. Kim, J. H. Park, S.-w. Hwang, and J. H. Cheon, "Privacy-preserving text classification on BERT embeddings with homomorphic encryption," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 3169–3175. [Online]. Available: https://aclanthology.org/2022.naacl-main.231