2024 13th International Conference on Control, Automation and Information Sciences (ICCAIS) | 979-8-3315-4204-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCAIS63750.2024.10814364

BERT-Enhanced DGA Botnet Detection: A Comparative Analysis of Machine Learning and Deep Learning Models

Qui Cao[1,2],Phuc Dao-Hoang[1,2], Dat-Thinh Nguyen[1,2], Xuan-Ha Nguyen[1,2], Kim-Hung Le[1,2]

[1]University of Information Technology, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
Email: {22521208, 22521110, 19520982}@gm.uit.edu.vn, {hanx, hunglk}@uit.edu.vn

***Abstract*—With the proliferation of Internet of Things, detecting Domain Generation Algorithm (DGA) botnets is critical for protecting networks from evolving and sophisticated cybersecurity threats. This paper explores a novel approach combining BERT with machine learning and deep learning techniques to detect DGA botnets. We provide a comprehensive benchmark by evaluating the performance of various BERT versions and detection methods on diverse datasets. Our experimental results reveal the significant impact of BERT version selection on detection accuracy and the superior performance of deep learning models, such as CNN, MLP, and LSTM, compared to conventional machine learning models. These findings highlight the potential of BERT and deep learning in improving DGA botnet detection and offer valuable insights for future research in this area.**

***Index Terms*—Domain Generation Algorithm botnet, Machine learning, Deep learning, BERT**

I. Introduction

A botnet is a network of compromised devices controlled by cybercriminals, known as botmasters, to carry out illegal activities without the owner's permission. These networks often include various networking devices, all controlled through malware typically delivered via phishing or exploiting outdated software vulnerabilities. Botnets pose a significant threat due to their scale and capacity, with examples like the Conficker and Mirai botnets [1], which infected millions of devices and were used for spamming, data theft, and DDoS attacks. The botmasters can control the infected devices, making detection and dismantling difficult. More advanced botnets, however, employ even greater evasion techniques.

One such variant is the Domain Generation Algorithm (DGA) botnet [2], which generates new domain names algorithmically to maintain contact with its C&C server. This enables the botnet to evade domain blacklisting and continue operations even if certain domains are blocked. By monitoring DNS traffic, security systems attempt to distinguish between benign and malicious domains, but the adaptive nature of DGA botnets makes them particularly challenging to neutralise. A prime example is the Conficker botnet [3], which used DGA to support large-scale spam and malware distribution, affecting millions of systems globally.

Detecting DGA botnets is increasingly difficult due to their ability to generate numerous domain names. Traditional methods struggle with these dynamic domains, leading to the adoption of machine and deep learning-based solutions. Machine learning models, such as decision trees and random forests, rely on features like domain structure and entropy, while deep learning models, including RNNs and CNNs, learn patterns from raw domain data. These approaches have shown promise in improving detection accuracy, particularly for previously unseen DGA families.

To further enhance DGA botnet detection, a promising solution is to combine a language model, such as BERT (Bidirectional Encoder Representations from Transformers), with a machine learning or deep learning model. BERT serves as a tokenizer, extracting meaningful features from domain names by capturing both lexical and contextual information. These features are then fed into a detection model, such as RF, CNN or RNN, to classify domains as benign or DGA-related. This hybrid approach leverages BERT's ability to process textual patterns and the deep learning model's classification precision, significantly boosting detection accuracy.

Despite advancements in DGA detection, the integration of BERT with the detection model has not been comprehensively explored. Furthermore, there is a lack of benchmarking studies comparing various BERT versions in this context. This paper addresses these gaps by introducing a novel approach and performing extensive experiments to benchmark the performance of different BERT and deep learning model combinations. The main contributions of this paper are as follows:

- We propose a DGA botnet detection approach that combines BERT with machine learning and deep learning-based model, enabling more effective feature extraction and improved classification accuracy.
- We conduct extensive benchmarking experiments, comparing various BERT versions and detection models, to provide a detailed analysis of their performance detecting DGA botnets. This benchmarking serves as a valuable resource for future research in the field.

979-8-3315-4204-7/24/$31.00 ©2024 IEEE

The remainder of this study is organized as follows: Section II provides some related works. Next, Section III presents in detail our methodology. Then, Section IV provides the experiment results and discussion. Finally, Section V presents the conclusion for this research.

II. Related work

DGA botnets represent one of the most challenging and elusive cybersecurity threats today. These botnets use domain generation algorithms to create random domains, making detection and blocking efforts more difficult. To address this threat, numerous studies have proposed various detection and prevention methods, including behavioural analysis, machine learning models, and algorithm-based detection techniques [4], [5]. This section presents an overview of related research, highlighting the techniques developed to effectively identify and mitigate DGA botnets.

A. Domain generation algorithm

Ranjana B. Nadagoudar and M. Ramakrishna [6] compared the performance of LSTM, RNN, and GRU models in distinguishing legitimate and malicious domain names, specifically focusing on their ability to detect computer-generated domain names. Their study demonstrated that the GRU model, which achieved up to 99% accuracy in binary classification, is highly effective in detecting and classifying DGA domains, marking a significant advancement in network security. Yu Fu et al. [7] proposed two new domain generation algorithms based on HMM and PCFG and evaluated their ability to evade detection compared to existing algorithms. The experimental results showed that these algorithms exhibit better detection evasion capabilities than current alternatives. Similarly, Hieu Mac et al. [8] examined the effectiveness of various machine learning methods for detecting DGA botnets using a large real-world dataset. Their comparison included popular methods such as C4.5, ELM, and SVM, along with newer approaches like LSTM, BiLSTM, and Recurrent SVM. LSTM and BiLSTM achieved over 90% accuracy, demonstrating strong performance in identifying DGA patterns, while the other methods also showed promising results.

B. DGA botnet detection

Anwar. S. R et al. [9] proposed an effective deep learning framework to help detect malware using the DGA algorithm to generate domain names. The authors proposed a two-level model that combines feature generation and block clustering and classification. They use seven linguistic features to describe domain names. MLP classifier helps distinguish between DGA and normal, K-means clusters that group related DGA. The results show that the model achieves high accuracy, surpassing other methods. The article has achieved its intended purpose and contributed to improving malware detection. Ahmed M. Manasrah et al. [10] proposed a method for detecting DGA botnets using machine learning techniques to classify domains as legitimate or illegitimate. Their approach focuses on evaluating linguistic features extracted from domain names in DNS request packets to identify DGA-based botnets. By testing their method on multiple real-world datasets, both simulated and actual data, they achieved a detection rate of 99.1% with a false positive rate of 0.6%. Yong-lin Zhou et al. [11] introduced a new method for detecting DGA botnets by analysing DNS protocol queries, explicitly focusing on non-existing domain (NXDomain) queries. Although their method successfully filtered out potential DGA domains, it required more time to accurately confirm suspicious domains, highlighting the need for future improvement.

III. Methodology

A. Overview

In our research, we aim to evaluate the performance of a detection system which combines BERT as a tokenizer and a deep learning model as a classifier. Hence, we conduct a comprehensive experiment with different BERT versions and various deep-learning models. The detection system follows a structured workflow, beginning with Preprocessing and culminating in a comprehensive Detection Model evaluation.

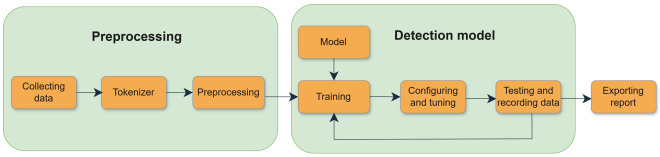

Fig. 1. Our workflow on evaluating a detection model.

- **Preprocessing:** In this stage, relevant datasets are collected from various sources that are suitable for the detection task. The collected data is cleaned to ensure its quality and usability. This involves preparing labels for classification, removing duplicate and invalid samples, and addressing inconsistencies in the dataset. To convert domain names into numerical features suitable for machine learning, BERT is applied as a tokenizer. BERT captures both lexical and contextual features, transforming raw text into a structured format that the detection model can process efficiently.
- **Building the detection model:** After preprocessing, the dataset is split into training and testing subsets. The detection model is then trained and evaluated. Throughout the training phase, the model's hyperparameters are tuned to achieve the best configuration for detection accuracy. Once the model is trained, it is tested on the unseen test data to evaluate its performance. The final step involves recording and exporting a detailed report highlighting the model's effectiveness in detecting DGA botnets.

B. Bidirectional Encoder Representations from Transformers

BERT [12] is a language model developed by Google that captures the full context of words by processing text in both

directions rather than traditional models that read the text one way. This makes BERT highly effective for different tasks. Our detection model uses BERT as a tokenizer to extract features from domains. To convert a domain into a token vector, BERT tokenizes input text into subword units, mapping them to numerical IDs from a predefined vocabulary, with special tokens like [CLS] and [SEP] added. Each token is then transformed into a fixed-length vector and fed into a detection model to classify domains. In this study, we used three state-of-the-art BERT versions, described below:

- **LinkBERT** [13] is an improved version of BERT that helps to cover the relationship between documents through hypertext links and citations. It allows to connect knowledge from different sources. This model can directly replace BERT in many language applications. LinkBERT gives better results especially in multi-text understanding and deep knowledge tasks.
- **DistilBERT** [14] is a scaled-down and faster version of the original BERT. It is also pre-trained on the same dataset as BERT, but uses a self-supervised training method, using the BERT base model itself as a teacher. This means that DistilBERT is trained only on the text directly without humans labeling them, and uses the automated process to generate data samples and labels from BERT. DistilBERT is a scaled-down version of BERT that retains language understanding.
- **CodeBERT** [15] is a model specifically designed for trainers to understand language implementers. The special feature of CodeBERT is that it is trained on both natural language and source code of 6 popular programming languages, including Python, Java, Javascript, PHP, Ruby, and Go. Therefore, CodeBERT can understand the relationship between natural language and code instructions, helping to apply in many tasks related to installers and software development.

C. The detection model

In this paper, we evaluate the detection performance of different widely used machine learning and deep learning models. Particularly, our experiments include three machine-learning models and four deep-learning models as below:

Deep Learning Models:

- **Convolutional Neural Networks (CNNs)** are a powerful deep learning architecture, particularly effective in tasks involving computer vision and image processing. CNNs are a specialised class of neural networks designed to process grid-structured data efficiently, making them highly suitable for visual data.
- **Multi-Layer Perceptions (MLPs)** are an artificial neural network composed of multiple layers of neurons. These neurons typically utilise nonlinear activation functions, enabling MLPs to capture complex patterns in data. This ability to learn nonlinear relationships makes MLPs particularly useful for tasks such as classification, regression, and pattern recognition.
- **Recurrent Neural Networks (RNNs)** are deep learning models that process sequential data, such as text, sentences, or time series. RNNs are structured to handle temporal dependencies, similar to how humans interpret sequential data, such as in language translation.
- **Long Short-Term Memory Networks (LSTMs)**, introduced by Hochreiter and Schmidhuber, are an advanced version of RNNs. LSTMs are specifically designed to capture long-term dependencies in sequential data, making them highly effective for tasks like language translation, speech recognition, and time series forecasting.

Machine Learning Models:

- **Random Forest** is an ensemble algorithm that combines multiple decision trees, each built from random subsets of data and features. Predictions are made by voting (for classification) or averaging (for regression) across all trees. This randomness reduces overfitting and enhances model robustness, making Random Forest effective for complex datasets.
- **Support Vector Machine (SVM)** is a supervised machine learning algorithm primarily used for classification tasks. It works by finding the optimal hyperplane that maximises the margin between different classes in a dataset, ensuring that the data points from each class are as far apart as possible.
- **K-Nearest Neighbors (KNN)** is a fundamental classification algorithm widely used in pattern recognition, data mining, and intrusion detection. As a supervised learning method, KNN operates on the principle that data points with similar features are likely to share the same labels. During the training phase, KNN stores the entire dataset for reference when classifying new instances based on proximity to its nearest neighbours.

D. Datasets

In this paper, we use four distinct datasets to evaluate the performance of our proposed detection model. These datasets encompass a mix of benign and malicious domain names, providing a diverse foundation for training and testing the detection model's ability to differentiate between regular domain traffic and domains generated by DGA botnets. The detail description is listed below:

- **majestic_million.csv** [16] includes 1,101,673 samples belonging to 70 classes (1 legitimate class and 69 DGA classes). The normal domains are from the Majestic top 1 million dataset, while the malicious domains are obtained from the DGArchive.
- **argencon.csv** [1] consists of 2,918,496 samples, distributed across 53 classes—2 legitimate classes and 51 DGA classes. The legitimate domain names were sourced from the Alexa Top 1 Million domains, while

[1] https://huggingface.co/datasets/harpomaxx/dga-detection

the DGA domain names were obtained from the repositories curated by Andrey Abakumov and John Bambenek.

- **umbrella_million.csv** [2] includes 2,324,298 samples belonging to 49 classes (1 legitimate class and 48 DGA classes). The benign domains are from the Majestic top 1 million dataset, while the malicious domains are obtained from Netlab 360.
- **dga_data.csv** [3] includes 160,000 samples belonging to 9 classes (2 legitimate classes and 7 DGA classes). This dataset has been collected from Alexa website ranking and a blacklist of previous DGA domain names.

E. Evaluation metrics

In this paper, to evaluate the performance of our proposed detection model, we use four key evaluation metrics: Accuracy, F1 Score, Recall (True Positive Rate), and Precision (Positive Predictive Value). These metrics are derived from the confusion matrix, which is based on four essential components: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

IV. Experimental Results

This section presents the comprehensive results of our experiments, focusing on two key tasks: binary classification and multiclass classification. For each task, we evaluate the model's performance using four mentioned metrics. The best score in each metric within each BERT version is highlighted in red, while the second-best score is marked in blue.

A. Binary classification results

TABLE I
BINARY CLASSIFICATION RESULTS OF DGA_DATA DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	**Model**				
LinkBERT	CNN	86.80	87.17	90.73	84.01
	MLP	86.81	86.81	86.81	87.01
	RNN	84.80	84.20	80.77	88.17
	RDF	92.03	91.62	86.74	97.08
	SVM	91.51	91.05	85.94	96.80
	KNN	92.01	91.96	90.98	92.95
	LSTM	85.21	84.97	83.92	86.26
DistilBERT	CNN	87.54	87.77	91.25	84.67
	MLP	87.52	87.52	87.52	87.71
	RNN	84.94	84.55	83.66	85.68
	RDF	91.87	91.49	87.11	96.34
	SVM	91.26	90.81	86.00	96.20
	KNN	92.67	92.63	91.80	93.48
	LSTM	84.84	83.94	80.79	87.56
CodeBERT	CNN	86.54	86.20	85.00	87.62
	MLP	85.71	85.71	85.71	85.92
	RNN	83.84	83.17	80.85	85.86
	RDF	91.74	91.31	86.45	96.76
	SVM	89.70	89.07	83.54	95.38
	KNN	87.18	86.30	80.39	93.14
	LSTM	86.91	86.89	86.91	87.33

[2]https://www.kaggle.com/datasets/xeric7/dga-detection
[3]https://www.kaggle.com/datasets/gtkcyber/dga-dataset

Tables I, II, III, and IV, respectively, present the binary classification results of four benchmarked datasets. The tables generally show that the proposed solution achieves high accuracy in detecting DGA botnets, with the best accuracy of each dataset being above 91%. Additionally, the results indicate the significant impact of choosing the BERT version on detection performance. Furthermore, it also shows that the CNN and MLP models stably achieve a higher accuracy than other models. In more detail:

TABLE II
BINARY CLASSIFICATION RESULTS OF ARGENCON DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	**Model**				
LinkBERT	CNN	92.89	92.78	92.14	93.55
	MLP	91.64	91.62	91.96	91.40
	RNN	79.73	88.31	82.78	85.02
	RDF	87.94	87.40	83.75	91.37
	SVM	82.92	81.77	76.72	87.53
	KNN	89.62	89.44	88.06	90.87
	LSTM	90.77	90.66	90.39	91.06
DistilBERT	CNN	92.11	91.97	91.07	93.02
	MLP	91.67	91.49	90.20	92.94
	RNN	84.26	82.85	76.65	90.43
	RDF	87.14	86.66	83.67	89.86
	SVM	83.19	82.42	78.93	86.23
	KNN	89.38	89.25	88.27	90.25
	LSTM	84.29	82.90	76.78	90.36
CodeBERT	CNN	91.72	91.67	91.73	91.74
	MLP	82.11	83.64	92.16	76.72
	RNN	83.09	82.87	82.61	83.36
	RDF	86.66	86.14	83.03	89.49
	SVM	76.13	73.47	66.22	82.51
	KNN	85.29	84.62	81.09	88.47
	LSTM	90.26	90.23	90.70	89.91

TABLE III
BINARY CLASSIFICATION RESULTS OF MAJESTIC MILLION DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	**Model**				
LinkBERT	CNN	91.32	91.34	92.10	90.70
	MLP	90.75	90.76	91.49	90.17
	RNN	79.81	78.08	72.79	84.49
	RDF	82.63	80.58	71.85	91.73
	SVM	83.26	81.79	74.96	89.98
	KNN	87.72	87.57	86.24	88.95
	LSTM	82.22	79.75	70.98	91.39
DistilBERT	CNN	91.12	90.97	91.03	91.03
	MLP	89.82	89.67	89.14	90.36
	RNN	79.48	76.36	66.63	89.85
	RDF	82.69	80.59	71.62	92.12
	SVM	81.55	79.82	72.79	88.36
	KNN	86.34	86.07	84.14	88.09
	LSTM	79.54	76.34	66.63	89.76
CodeBERT	CNN	86.85	86.73	86.52	87.16
	MLP	82.27	83.34	89.56	78.09
	RNN	75.70	73.97	69.47	79.46
	RDF	80.68	78.96	72.29	86.98
	SVM	79.77	78.17	72.22	85.20
	KNN	81.83	80.90	76.72	85.55
	LSTM	83.10	83.29	84.92	81.93

The experiment results of four datasets show that choosing the BERT version has an impact on the detection performance. The LinkBERT give the best accuracy on three out

TABLE IV
BINARY CLASSIFICATION RESULTS OF UMBRELLA_MILLION DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	Model				
LinkBERT	CNN	96.99	96.96	96.70	97.27
	MLP	96.57	96.57	96.57	96.63
	RNN	51.26	51.37	51.93	51.21
	RDF	93.11	93.01	91.73	94.32
	SVM	72.69	72.55	72.27	72.82
	KNN	90.88	90.96	91.92	90.02
	LSTM	50.08	66.63	49.96	50.08
DistilBERT	CNN	96.88	96.87	96.89	96.89
	MLP	96.29	96.29	96.29	96.36
	RNN	60.48	56.86	52.32	62.78
	RDF	92.96	92.86	91.63	94.11
	SVM	73.99	73.87	73.64	74.10
	KNN	90.30	90.30	90.34	90.25
	LSTM	49.92	66.48	50.32	49.92
CodeBERT	CNN	96.44	96.42	96.56	96.34
	MLP	96.74	96.20	96.91	96.26
	RNN	50.08	66.62	49.27	50.08
	RDF	93.29	93.12	90.96	95.40
	SVM	65.77	63.38	59.32	68.03
	KNN	86.99	86.51	83.54	89.70
	LSTM	50.04	66.58	49.87	50.04

of four datasets, while the CodeBERT performed worst compared to other BERT versions. Additionally, the LinkBERT and DistilBERT maintain a stable high accuracy across four datasets, while the CodeBERT perform worst at the Majestic Million dataset with the best accuracy of 86.85% compared to 91.32% of the LinkBERT. In addition to the BERT versions, the CNN and MLP models stand out for their superior detection performance. They achieve the top accuracy scores in three out of four datasets, except for DGA_Data, where KNN achieves the best result. CNN and MLP also perform well in F1, TPR, and PPV metrics, further solidifying their effectiveness.

In conclusion, it shows that selecting the BERT version impacts detection accuracy, with LinkBERT consistently outperforming other versions. Additionally, CNN and MLP models emerge as the best-performing models across most datasets, making them the optimal choices for DGA botnet detection tasks. This demonstrates the importance of both model and BERT selection in achieving high detection performance.

B. Multiclass classification results

Tables V, VI, VII and VIII present the multiclass classification results for four datasets: DGA_Data, Argencon, Majestic Million, and Umbrella Million. The tables indicate that the proposed solution achieves acceptable accuracy in classifying DGA botnets, with the best accuracy reaching approximately 80% in three out of four datasets. The results highlight the significant impact of selecting the BERT version on classification performance. Furthermore, they reveal that deep learning models consistently outperform other models.

The findings demonstrate that while the BERT version does affect detection performance, no single version significantly outperforms the others. LinkBERT achieves the best

TABLE V
MULTICLASS CLASSIFICATION RESULTS OF DGA_DATA DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	Model				
LinkBERT	CNN	60.97	55.74	60.97	59.56
	MLP	60.64	55.72	60.64	58.50
	RNN	50.31	42.22	50.31	39.90
	RDF	57.24	51.90	56.08	60.69
	SVM	48.75	46.67	50.78	50.06
	KNN	52.92	47.82	48.07	55.61
	LSTM	50.77	41.50	50.77	40.74
DistilBERT	CNN	59.80	56.14	59.80	61.64
	MLP	61.35	57.24	61.35	60.64
	RNN	51.07	46.89	51.07	47.89
	RDF	58.03	52.63	56.75	54.41
	SVM	52.06	50.24	54.93	53.50
	KNN	54.18	49.42	49.62	57.14
	LSTM	51.51	46.41	51.51	49.03
CodeBERT	CNN	58.41	53.72	58.41	53.57
	MLP	55.11	50.48	55.11	52.16
	RNN	48.26	38.83	48.26	38.48
	RDF	53.59	42.18	46.70	51.36
	SVM	42.88	34.65	37.92	41.38
	KNN	43.06	36.69	37.68	38.93
	LSTM	58.79	53.91	58.79	56.83

accuracy in two of the four datasets, while DistilBERT and CodeBERT each record the best performance in one dataset. All BERT versions maintain high accuracy in three datasets, except for the DGA_Data dataset, likely due to its complex structure and class imbalance across 70 classes. Furthermore, CNN, MLP, and LSTM models distinguish themselves with superior detection performance, achieving the top accuracy in all datasets. Specifically, CNN records the highest accuracy in the Argencon and Umbrella Million datasets, MLP leads in the DGA_Data dataset, and LSTM achieves the best result in the Majestic Million dataset.

TABLE VI
MULTICLASS CLASSIFICATION RESULTS OF ARGENCON DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	Model				
LinkBERT	CNN	83.70	83.38	83.70	84.74
	MLP	81.50	80.91	81.50	81.81
	RNN	40.97	37.25	40.97	39.55
	RDF	62.10	57.08	59.91	61.94
	SVM	49.10	43.89	46.27	52.46
	KNN	74.59	72.57	73.18	73.42
	LSTM	45.59	41.90	45.59	43.34
DistilBERT	CNN	83.12	82.80	83.12	83.90
	MLP	82.10	81.75	82.10	82.17
	RNN	39.75	34.82	39.75	37.64
	RDF	63.11	59.32	60.50	67.34
	SVM	46.71	40.34	44.33	43.34
	KNN	72.70	70.17	70.89	70.97
	LSTM	40.30	35.56	40.30	38.57
CodeBERT	CNN	80.29	79.66	80.29	81.46
	MLP	75.54	74.52	75.54	75.31
	RNN	39.94	32.72	39.94	31.54
	RDF	57.27	52.39	54.17	64.61
	SVM	38.12	32.25	36.54	33.73
	KNN	59.36	56.25	57.15	56.74
	LSTM	76.01	75.80	76.01	77.54

In conclusion, the choice of BERT version significantly

TABLE VII
MULTICLASS CLASSIFICATION RESULTS OF MAJESTIC MILLION DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	**Model**				
LinkBERT	CNN	78.80	77.16	78.80	79.15
	MLP	78.67	77.00	78.67	79.35
	RNN	53.68	47.09	53.68	46.79
	RDF	63.14	41.70	45.41	44.99
	SVM	53.13	32.89	35.48	36.07
	KNN	63.32	46.20	46.66	47.42
	LSTM	92.69	90.52	92.69	88.77
DistilBERT	CNN	75.86	74.05	75.86	77.07
	MLP	76.42	74.00	76.42	74.95
	RNN	52.51	46.84	52.51	46.59
	RDF	65.20	41.18	45.24	52.52
	SVM	56.27	34.59	38.97	41.32
	KNN	63.28	47.58	48.37	48.70
	LSTM	91.98	89.34	91.98	87.27
CodeBERT	CNN	70.40	66.74	70.40	68.95
	MLP	70.18	67.38	70.18	68.44
	RNN	53.37	48.22	53.37	46.18
	RDF	63.61	40.96	43.98	43.17
	SVM	51.31	29.25	31.27	31.23
	KNN	56.13	37.15	38.71	37.86
	LSTM	90.75	86.37	90.75	82.43

TABLE VIII
MULTICLASS CLASSIFICATION RESULTS OF UMBRELLA MILLION DATASET (%)

Architecture		ACC	F1	TPR	PPV
BERT	**Model**				
LinkBERT	CNN	93.50	93.48	93.50	93.88
	MLP	91.69	91.61	91.69	91.88
	RNN	21.71	07.89	21.71	04.85
	RDF	93.11	93.01	91.73	94.32
	SVM	54.29	40.85	42.73	48.83
	KNN	79.78	77.85	78.87	77.25
	LSTM	21.72	07.91	21.72	04.86
DistilBERT	CNN	92.01	91.96	90.98	92.95
	MLP	94.62	94.59	94.62	94.77
	RNN	21.60	07.83	21.60	04.81
	RDF	93.06	92.93	91.31	94.61
	SVM	57.98	40.67	42.51	52.41
	KNN	84.02	81.31	81.75	81.15
	LSTM	21.73	07.91	21.73	04.86
CodeBERT	CNN	95.72	95.68	95.72	95.80
	MLP	80.72	79.63	80.72	81.77
	RNN	21.64	07.85	21.64	04.82
	RDF	93.29	93.12	90.96	95.40
	SVM	53.03	33.17	37.01	37.17
	KNN	77.32	73.03	73.65	73.24
	LSTM	21.68	07.86	21.68	04.83

affects classification accuracy, with LinkBERT, DistilBERT, and CodeBERT excelling on different datasets. CNN, MLP, and LSTM consistently achieved the highest accuracy, making them the most effective for DGA botnet detection. However, challenges like dataset complexity and class imbalance, especially in datasets like DGA_Data, can negatively affect detection performance.

V. CONCLUSION

This paper explores the use of BERT combined with machine and deep learning models for detecting DGA botnets. By benchmarking various BERT versions and detection methods across multiple datasets, we assessed the effectiveness of BERT in improving detection accuracy. The results show that selecting BERT version significantly impacts performance, with deep learning models consistently outperforming traditional machine learning models. These findings provide important insights for future research to enhance DGA botnet detection strategies.

VI. ACKNOWLEDGEMENT

This research is funded by University of Information Technology-Vietnam National University HoChiMinh City under grant number D1-2024-46.

REFERENCES

[1] Han Zhang, Manaf Gharaibeh, Spiros Thanasoulas, and Christos Papadopoulos. Botdigger: Detecting dga bots in a single network. In *TMA*, 2016.

[2] Arthur Drichel, Marc Meyer, and Ulrike Meyer. Towards robust domain generation algorithm classification. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, 2024.

[3] Tzy-Shiah Wang, Hui-Tang Lin, Wei-Tsung Cheng, and Chang-Yu Chen. Dbod: Clustering and detecting dga-based botnets using dns traffic analysis. *Computers & Security*, 64:1–15, 2017.

[4] Tong Anh Tuan, Hoang Viet Long, and David Taniar. On detecting and classifying dga botnets and their families. *Computers & Security*, 113:102549, 2022.

[5] Hoang-Cong-Thanh Nguyen, Xuan-Ha Nguyen, and Kim-Hung Le. An automated benchmarking framework for anomaly-based intrusion detection systems. In *2024 International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*, 2024.

[6] Ranjana B. Nadagoudar and M. Ramakrishna. Dga domain name detection and classification using deep learning models. *International Journal of Advanced Computer Science & Applications*, 15(7), 2024.

[7] Yu Fu, Lu Yu, Oluwakemi Hambolu, Ilker Ozcelik, Benafsh Husain, Jingxuan Sun, Karan Sapra, Dan Du, Christopher Tate Beasley, and Richard R. Brooks. Stealthy domain generation algorithms. *IEEE Transactions on Information Forensics and Security*, 2017.

[8] Hieu Mac, Duc Tran, Van Tong, Linh Giang Nguyen, and Hai Anh Tran. Dga botnet detection using supervised learning methods. In *Proceedings of the 8th International Symposium on Information and Communication Technology*, pages 211–218, 2017.

[9] M. B. Smithamol, Vinodu George, and Abdul Rahiman. A deep learning framework for domain generation algorithm-based malware detection. *Journal Name*, 2023.

[10] Ahmed M. Manasrah, Thair Khdour, and Raeda Freehat. Dga-based botnets detection using dns traffic mining. *Journal of King Saud University-Computer and Information Sciences*, 2022.

[11] Yonglin Zhou, Qing shan Li, Qidi Miao, and Kangbin Yim. Dga-based botnet detection using dns traffic. *Journal of Internet Services and Information Security*, 3(3/4):116–123, 2013.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Bidirectional encoder representations from transformers. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Michihiro Yasunaga, Jure Leskovec, and Percy Liang. Linkbert: Pretraining language models with document links. *arXiv preprint arXiv:2203.15827*, 2022.

[14] V Sanh. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[15] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*, 2020.

[16] Ibrahim Yilmaz, Ambareen Siraj, and Dennis Ulybyshev. Improving dga-based malicious domain classifiers for malware defense with adversarial machine learning. In *2020 IEEE 4th Conference on Information Communication Technology (CICT)*, Chennai, India, 2020. IEEE.