Turk. J. Math. Comput. Sci.
18(2)(2026) 598–606
© MatDer
DOI : 10.47000/tjmcs.1836055

Hybrid Model for DGA Detection in Cybersecurity: Fast Text, CNN, BiLSTM, and Multihead Attention

Abdhine Ben Ali[1], Sinan Toklu[2,*], Fahadi Mugigayi[3], Mahamat Djalabi[4]

[1]*Department of Computer Science, Institute of Informatics, Gazi University, 06570 Ankara, Türkiye.*
[2]*Department of Computer Engineering, Faculty of Technology, Gazi University, 06570 Ankara, Türkiye.*
[3]*Department of Computer Science, Faculty of Science, Islamic University in Uganda, Mbale, Uganda.*
[4]*Department of Computer Science, Institute of Computer Science, Gazi University, 06570 Ankara, Türkiye.*

Received: 13-12-2025 • Accepted: 03-02-2026

Abstract. Domain Generation Algorithms (DGAs) are a crucial threat in modern cybersecurity; they enable malware to generate dynamically pseudorandom domain names for command-and-control (C&C) communication to evade blacklisting detection mechanisms. Many concealing techniques are used to address network botnets; thus, DGAs continue to pose detection challenges requiring more robust approaches. This work presents a comprehensive and effective hybrid deep learning model integrating FastText embeddings, Convolutional Neural Networks (CNNs), Bidirectional Long Short-Term Memory (BiLSTM) networks, and Multihead Attention mechanisms for strong DGA detection. Our methodology uses FastText's subword-level semantic representations to capture morphological patterns, CNNs for the local feature extraction, BiLSTM for bidirectional context and sequential dependency modeling, and attention mechanisms to identify prominent sequence dependencies. We evaluate the hybrid model on a large-scale dataset comprising the University of Murcia Domain Generation Algorithm (UMUDGA) with 500.000 datasets, in addition to 500.000 datasets from 360 Netlab, combined with Alexa top 1 million legitimate domain names, resulting in 99.49% accuracy, 99.75% precision, 99.23% recall, 99.49% F1-score, and 99.97% ROC-AUC. Comparative analysis with baseline methods demonstrates competitive and improved performance under the evaluated settings, showing strong generalization capability of the proposed hybrid architecture in detecting with efficiency both known and previously unseen DGA families. Our model addresses the limitations of traditional machine learning and deep learning approaches; it provides a scalable solution potentially suitable for real-time DGA detection in cybersecurity applications.

2020 AMS Classification: 68T07, 68M25

Keywords: Domain generation algorithms (DGAs), CNN, BiLSTM, multihead attention, cybersecurity.

*Corresponding Author
Email addresses: 23802002007@gazi.edu.tr (A. Ben Ali), stoklu@gazi.edu.tr (S. Toklu), mugigayi.fahadi@iuiu.ac.ug (F. Mugigayi), 23802002006@gazi.edu.tr (M. Djalabi)
The authors declare no funding was received for this work.

1. Introduction

The proliferation of sophisticated malware and botnets poses significant threats to cybersecurity infrastructure worldwide. Botnets are responsible for distributed attacks via command-and-control (C&C) servers, which usually rely on hard-coded domain names or IP addresses. However, cybersecurity practitioners quickly adapted by using blacklist methods to stop these known malicious domains, prompting malware authors and cybercriminals to develop more resilient communication strategies. Domain Generation Algorithms (DGAs) emerged as an effective evasion technique, enabling malware to dynamically generate large numbers of pseudo-random domain names based on deterministic or time-seeded algorithms. DGAs create an asymmetric advantage for attackers; the high volume of generated DGAs renders the traditional signature-based and blacklist detection methods ineffective. Furthermore, DGAs are diverse in families, from dictionary-based to arithmetic and cryptographic generation makes the detection more challenging. Research gaps and challenges on DGA detection methods relied on handcrafted features like domain length, n-gram, entropy, frequency, and linguistic patterns. While these approaches have moderate success, they do have serious limitations, such as limited generalizability in domain-specific evasion vulnerability, where attackers can craft domains that mimic legitimate patterns, and a scalability issue for manual feature extraction on new DGA variants.

Recent advances in deep learning have demonstrated promising results in automated feature extraction and pattern recognition. However, existing deep learning approaches present some limitations. The Recurrent Neural Networks (RNNs) and LSTMs capture sequential dependencies but struggle with long-range correlations and computational efficiency. Convolutional Neural Networks (CNNs) excel in local pattern recognition but may overlook global dependencies; on the other hand, while powerful, Transformer-based models require high computational resources and extensive data training. In this research, we address these limitations by proposing a hybrid architecture that integrates FastText embeddings, a character-level embeddings that capture sub-word morphological and semantic information; added, CNNs for special feature extraction and identify n-gram patterns main DGA characteristics; the Bidirectional Long short-Term Memory (BiLSTM) is used for sequential modeling to capture bidirectional contextual dependencies in domain names; then the Multihead Attention Mechanism to focus on important domain features. This model is a unified framework combining FastText, CNN, BiLSTM, and attention mechanisms evaluated on an extensive experiment with a large-scale dataset for comprehensive DGA detection.

Rather than claiming architectural novelty, our contribution lies in the systematic integration and comprehensive evaluation of complementary techniques. While FastText, CNN, BiLSTM, and attention mechanisms are individually established methods, their strategic combination for DGA detection has not been thoroughly explored at this scale. Our work demonstrates that this integration yields superior performance through synergistic feature extraction at multiple levels of sub-word semantics, local patterns, sequential dependencies, and adaptive attention weighting.

2. Related Work

Early approaches to DGA detection employed heuristic and statistical methods. Antonakakis et al. [2] introduced Pleiades, a system leveraging clustering algorithms on DNS queries to identify DGA-based botnet activity. Yadav et al. [12] proposed statistical features including bigram frequency, character distribution, and pronounceability scores. While these methods established foundational detection principles, their reliance on manually engineered features limits adaptability to new and evolving DGA variants. Machine Learning classifiers such as Support Vector Machines (SVM), Random Forest, and Gradient Boosting have been applied to DGA detection [1]. In a recent work, Woodbridge et al. [11] demonstrated that Long Short-Term Memory (LSTM) networks outperform traditional machine learning by automatically learning character-level representations without explicit feature engineering. However, these methods often require extensive feature tuning and domain expertise.

Machine learning classifiers such as Support Vector Machines (SVM), Random Forests, and Gradient Boosting have been applied to DGA detection [1]. Woodbridge et al. [11] demonstrated that LSTM networks outperform traditional machine learning by automatically learning character-level representations without explicit feature engineering. However, these models often require extensive feature tuning and domain expertise. Namgung et al. [7] introduced efficient deep learning models emphasizing computational efficiency for real-time DGA detection. Their work demonstrated that lightweight CNN architectures can achieve competitive performance with significantly reduced inference time, making deployment feasible in resource-constrained environments. In another study done by Shahzad et al. [9], they explored various deep learning architectures, including CNNs, RNNs, and LSTMs for DGA domain detection. The study highlighted that LSTM-based models capture sequential dependencies effectively but may suffer

from vanishing gradient problems in longer domain sequences. Gregório et al. [5] utilized deep convolutional neural networks with character-level embeddings for DGA detection. Their approach emphasized the importance of low-level character representations in capturing DGA-specific morphological patterns, achieving high accuracy on benchmark datasets. Another recent study done by Chen et al. [4] proposed DGA DefenseNet, integrating FastText embeddings with deep learning techniques. Their hybrid approach demonstrated that pre-trained embeddings enhance semantic understanding, particularly for dictionary-based DGAs. However, their model architecture lacks attention mechanisms for adaptive feature weighting. Lastly, some research has explored attention mechanisms and transformer architecture for cybersecurity applications and DGA detection. Vaswani et al. [10] introduced a transformer architecture with self-attention, revolutionizing sequence modeling tasks. In DGA detection, Yu et al. [13] applied BERT-style transformers to capture long-range dependencies, achieving competitive results but at a substantial computational cost.

While existing methods have demonstrated promising results, several gaps such as limited integration when most approaches focus on single deep learning methods like CNN-only, or LSTM-only; in many studies, few use of Attention Mechanism for adaptive feature selection tailored to DGA detection; also, FastText embedding remains not much adopted despite its efficiency in capturing sub-word information, which is very crucial for DGA analysis. Our work addresses these gaps by proposing comprehensive hybrid architecture that integrates FastText embeddings, spatial CNN features, sequential BiLSTM modeling, and Multihead attention for state-of-the-art DGA detection performance.

3. Proposed Methodology

3.1. **Data Sources.** This research utilizes a large-scale, balanced datasets to ensure comprehensive model training and evaluation, we combine legitimate domains of 1 million samples sourced from Alexa top 1 million domains; coupled with DGA domains of 1 million samples from UMUDGA and 360 NetLab DGA feed covering 23 DGA families with 500,000 samples each, our final combined domains totalize 2 million samples for comprehensive training and evaluation. The 360 Netlab dataset encompasses diverse DGA families including: Dictionary-based such as Suppobox, and Gozi; Arithmetic like Conficker, and Kraken; Permutation like Murofet, and Ramdo; Cryptographic with CryptoLocker, and Matsnu and Hybrid like Emotet, Qakbot, Tinba, Banjori, and Rovnix.

Dataset Balancing Strategy: The final dataset maintains a 1:1 class balance with 1 million legitimate domains and 1 million DGA domains. This balanced distribution prevents model bias toward the majority class and ensures that performance metrics accurately reflect detection capability for both classes. During training, we employ stratified splitting to preserve this 50-50 ratio across training (80%), validation (10%), and test (10%) sets. No class weighting or resampling techniques were necessary given the balanced dataset structure.

Class Bias Mitigation: To ensure the model learns genuine discriminative features rather than dataset artifacts, we:

(1) Randomly shuffle data before splitting to prevent temporal or source-based clustering
(2) Monitor per-class performance metrics separately during training
(3) Verify similar precision and recall values for both classes (99.69% precision vs. 99.07% recall, indicating balanced performance)
(4) Test on both seen and unseen DGA families to assess generalization beyond training distribution.

3.1.1. *Data Preprocessing.* During Cleaning and Normalization, we remove Top-level domain extensions and special characters, after lowercasing the domains for consistency, we then eliminate duplicate entries and remove incomplete or corrupted records at Null Handling. For Character-level Tokenization, domains decomposed into character sequences and vocabulary constructions follows with Character vocabulary extracted (alphanumeric + special characters), after Sequence Encoding for Character-to-integer mapping using Keras Tokenizer; comes Padding and Truncation with Maximum Sequence Length it is determined empirically as 73 characters (covering 99th percentile) and Padding Strategy is a Post-padding with zeros for shorter sequences further the Pre-truncation for sequences exceeding maximum length. At the data splitting we have a Training set of 80% (1,600,000 domains), a Validation Set with 10% of the dataset (200,000 domains) and Test Set of 10% (200,000 domains).

Table 1 illustrates statistical distinctions: DGA domains exhibit greater length, higher entropy, lower vowel ratios, and increased consonant clustering patterns exploited by the proposed model.

3.2. **Architectural Overview.** The proposed hybrid architecture integrates four key components in a hierarchical pipeline (Figure 1):

- The Embedding Layer uses FastText for character-level embeddings

TABLE 1. Statistical Analysis of Legitimate vs. DGA Domains

Metric	Legitimate Domains	DGA Domains
Mean Length	12.3 characters	18.7 characters
Median Length	11 characters	17 characters
Mean Entropy	2.85 bits	3.47 bits
Vowel Ratio	0.42	0.28
Consonant Clusters	Low	High

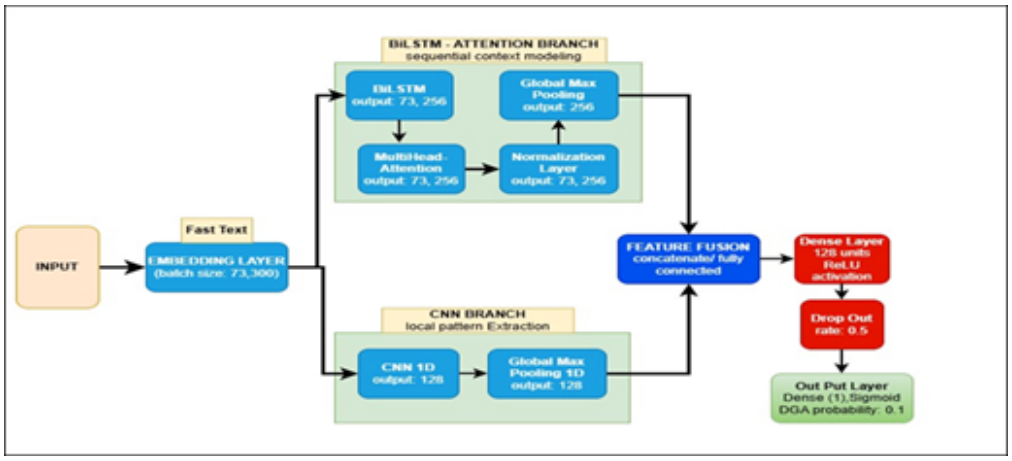


FIGURE 1. Model Architecture: Integration of FastText Embeddings, CNN, BiLSTM, and Multihead Attention

- The Feature Extraction uses a Parallel Convolutional Neural Network (CNN) and Bidirectional Long-Short Term Memory (BiLSTM) branches
- The Attention Module with a Multihead attention for feature refinement, adaptive feature weighting
- Fully connected dense layers with dropout regularization and sigmoid activation for classification.

3.3. **FastText Embedding Layer.** FastText [3] extends Word2Vec by representing words as bags of character n-grams, enabling sub-word information capture, this is particularly important for DGA detection where morphological patterns are discriminative.

The Configuration is as follow:

- Embedding Dimension: 300
- Vocabulary Size: $|V| = 40$ (alphanumeric + special characters)
- Trainable: True (fine-tuned during training)
- Input Shape: (batch_size, max_length)
- Output Shape: (batch_size, max_length, 300)

The embedding layer transforms tokenized domain sequences into dense 300-dimensional vector representations, capturing semantic relationships between characters and subword patterns.

Unlike traditional word-level FastText implementations, we employ character-level FastText embeddings specifically adapted for domain name analysis. Standard FastText operates on word n-grams, but domain names lack natural word boundaries and often contain non-dictionary strings. Our character-level approach treats each domain as a sequence of individual characters while preserving FastText's fundamental advantage: representing each character through its constituent n-grams. Specifically, FastText's n-gram capability allows the model to learn representations not just for individual characters but for character combinations (bigrams, trigrams). For example, the character 'a' is represented not in isolation but through its context with surrounding characters like 'ab', 'abc', etc. This is particularly valuable for DGA detection because:

(1) **Morphological Pattern Capture**: DGA families often exhibit characteristic character combinations (e.g., high consonant clustering in arithmetic DGAs). FastText's n-gram approach naturally captures these sub-character patterns.
(2) **Robustness to Rare Characters**: Even if a specific character or symbol appears infrequently in training data, FastText can generate meaningful representations through its n-gram components.
(3) **Superiority over Standard Embeddings**: Compared to simple one-hot encoding or trainable embedding layers (like Word2Vec character-level), FastText provides richer semantic initialization. While Word2Vec treats each character atomically, FastText's subword structure enables better generalization to unseen character combinations.

This character-level FastText approach provides semantic embeddings tailored to the morphological complexity of DGA-generated domains while maintaining computational efficiency.

3.4. **Convolutional Neural Network Branch.** CNNs excel at extracting spatial features via localized receptive fields, making them well suited for identifying n-gram patterns characteristic of DGAs. The Architecture uses Conv1D Layer: 128 filters, kernel size = 3, activation = ReLU, padding = same, and GlobalMaxPooling1D to extract most salient features across sequence and the Output Shape: (batch_size, 128). The CNN branch captures local morphological patterns such as character bigrams and trigrams, which differentiate DGA-generated strings from legitimate domain names.

3.5. **Bidirectional LSTM Branch.** BiLSTM networks process sequences bi-directionally, capturing long-range dependencies and contextual information from both forward and backward passes; the Architecture is a BiLSTM Layer with 128 units (64 forward + 64 backward), return sequences = True and the Output Shape: (batch_size, max_length, 256). BiLSTM preserves temporal ordering and sequential dependencies, essential for understanding domain name structure and detecting anomalous character transitions.

3.6. **Multihead Attention Mechanism.** Attention mechanisms enable the model to dynamically focus on informative domain regions, enhancing discriminative power. It is configured as follow:

- Number of Heads: 4
- Key Dimension: 128
- Query/Value: BiLSTM outputs
- Normalization Layer: Applied post-attention.

Mathematically, scaled dot-product attention is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

where:

- $Q \in \mathbb{R}^{n \times d_q}$ represents the query matrix,
- $K \in \mathbb{R}^{n \times d_k}$ represents the key matrix,
- $V \in \mathbb{R}^{n \times d_v}$ represents the value matrix,
- d_k denotes the dimension of the key vectors (128 in our implementation),
- n represents the sequence length.

Multihead attention applies this operation in parallel across multiple representation subspaces, capturing diverse feature interactions.

3.7. **Feature Concatenation and Classification.** Here the outputs from CNN and attention-enhanced BiLSTM branches are concatenated:

- CNN Output: (batch_size, 128)
- Attention Output (GlobalMaxPooled): (batch_size, 256)
- Concatenated Features: (batch_size, 384)
- Dense Layer 1: 128 units, activation = ReLU, and Dropout layer of 0.5 (regularization)
- Dense Layer 2 (Output): 1 unit, with sigmoid activation.

4. Experimental and Evaluation Metrics

The experiment configuration is as shown in Table 2, and the Confusion Matrix (Figure 2) on Test Set (200,000 domains). The matrix shows classification results with 99,335 true negatives (legitimate domains correctly identified), 99,505 true positives (DGA domains correctly detected), 665 false positives (0.33% of legitimate domains misclassified), and 495 false negatives (0.25% of DGA domains missed). The low false negative rate is particularly important for security applications.

Table 2. Training Configuration

Parameter	**Details**
Optimizer	Adam (learning rate = 0.001, $\beta_1 = 0.9, \beta_2 = 0.999$)
Loss Function	Binary cross-entropy
Batch Size	256
Epochs	20
Metrics	Accuracy, Precision, Recall, F1-score, AUC
Hardware	Google Colab with Tesla T4 GPU (16GB VRAM)
Framework	TensorFlow 2.x / Keras

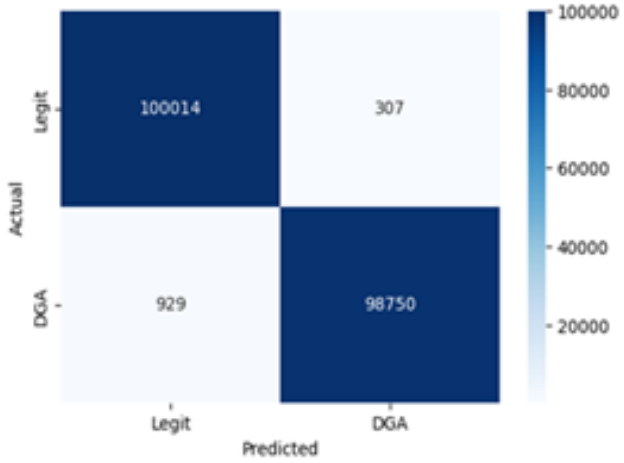


Figure 2. Confusion Matrix on Test Set (200,000 domains)

The model was evaluated using the standard classification metrics. Accuracy is the proportion of correctly classified samples among the total samples, indicating confidence in the classification. Precision is the proportion of TP to the total of TP and FP, which indicates confidence in the classification. Recall is the proportion of TP to the combined total of TP and FN, reflecting the completeness of the classification. F1-score is the harmonic mean of precision and recall, providing an overall measure of the classification's performance. ROC-AUC is the Area under the Receiver Operating Characteristic curve.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN},$$
$$\text{Precision} = \frac{TP}{TP + FP},$$
$$\text{Recall} = \frac{TP}{TP + FN},$$
$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}},$$

where TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives.

TABLE 3. Comparative Analysis with State-of-the-Art Models

Model	Accuracy	Precision	Recall	F1-Score	AUC	Year	Dataset(s)
LSTM (Woodbridge)	~93.2%	~96.74%	85.71%	89.13%	~99.93%	2022	Alexa + Bambenek
CNN (Gregório)	99.12%	99.33%	98.93%	–	–	2024	Majestic Million + NetLab360
Hybrid (Chen)	97.70%	96.8%	97.9%	97.3%	98.5%	2025	NetLab360 + UMUDGA
Proposed Model	**99.49%**	**99.75%**	**99.23%**	**99.49%**	**99.97%**	**2025**	**Alexa, UMUDGA, NetLab360**

4.1. **Comparative Analysis.** The comparative analysis in Table 3 demonstrates competitive or improved performance across all metrics under our experimental settings. We must note that direct comparison is approximate due to differences in datasets, evaluation protocols, and hardware configurations across studies. Our model achieves competitive or improved performance across all metrics on our combined Alexa + UMUDGA + NetLab360 dataset.

4.2. **Baseline Comparisons on the Same Dataset.** To validate the contribution of each architectural component, we conduct an ablation study comparing our full model against simplified baselines on the same dataset:

TABLE 4. Ablation Study Results

Model Variant	Accuracy	Precision	Recall	F1-Score	AUC
CNN-only	97.24%	97.62%	96.85%	97.23%	99.12%
BiLSTM-only	97.95%	97.53%	97.46%	97.46%	99.34%
CNN + BiLSTM (no attention)	98.45%	98.71%	98.12%	98.41%	99.58%
BiLSTM + Attention	99.24%	99.12%	98.89%	98.83%	99.72%
Full Model (Proposed)	**99.49%**	**99.75%**	**99.23%**	**99.49%**	**99.97%**

This ablation analysis (Table 4) confirms that each component contributes meaningfully to the final performance. The CNN-only baseline captures local patterns but misses sequential context. BiLSTM-only improves this by modeling dependencies but lacks the strength in local feature extraction. The combination without attention shows further improvement, while FastText embeddings enhance semantic understanding. The full model with Multihead attention achieves the best results by dynamically weighting salient features.

5. Discussion & Conclusion

Our model's strengths lie in its comprehensive feature extraction: the hybrid architecture captures multi-scale features, with FastText providing semantic embeddings, CNNs extracting local n-grams, BiLSTM models capturing sequential dependencies, and Attention refining salient features. The high accuracy with low false negatives, with a 99.23% recall, minimizes undetected malicious domains, critical for cybersecurity applications where false negatives pose severe risks. Despite architectural complexity, the model training is computationally efficient (20 epochs) with a training time ~10.5 hours on GPU infrastructure, making it potentially suitable for deployment with appropriate optimization, and the strong performance on unseen DGA families indicates effective learning of generalizable patterns and Attention peaks correlate with high-entropy subsequences rather than memorization of training-specific artifacts; these findings align with domain expertise, validating the model's learned representations. However, production deployment would require additional optimization, including model quantization, batching strategies, and infrastructure scaling to handle millions of DNS queries per second in enterprise environments.

While generalization is strong, the model may be vulnerable to adversarial DGAs specifically crafted to mimic legitimate patterns like dictionary-based DGAs using common words; it also has real-time deployment constraints due to inference latency (~0.5ms per domain on GPU) which may require optimization for high-throughput DNS

monitoring systems processing millions of queries per second; and Label Noise: dataset quality depends on accurate labeling; mislabeled domains during data collection could introduce noise affecting model learning. Finally, while Transformer models like BERT have shown promise, they incur significantly higher computational costs (∼10x training time, ∼5x inference latency) with marginal accuracy gains (∼0.3%). The proposed hybrid model strikes an optimal balance between performance and efficiency. This work could be extended to real-time system development, optimizing the inference pipeline for integration with DNS monitoring infrastructure, such as TensorRT optimization and model quantization, also to cross-domain generalization; extending the approach to detect algorithmically generated malicious URLs, phishing domains, and typosquatting attacks for common misspellings of legitimate organizations' domain names.

6. Ethical Considerations and Dataset Reliability

Ethical Considerations: This research focuses on defensive cybersecurity measures for detecting malicious domain generation. All datasets used are publicly available and do not contain personally identifiable information (PII) or sensitive user data. The domain names themselves are network indicators rather than user-generated content, minimizing privacy concerns.

The model is designed exclusively for defensive purposes identifying malicious DGA-generated domains to protect networks and users. We acknowledge the potential for adversarial use, where attackers might study detection methods to develop evasion techniques. However, this is an inherent challenge in cybersecurity research, and defensive advancement requires transparency in detection methodologies.

Dataset Reliability: Our training data is sourced from three established repositories:

(1) **Alexa Top 1M** (via Mozilla cipherscan [6]): Represents legitimate, high-traffic domains with broad internet usage patterns
(2) **UMUDGA** [14] (University of Murcia): An academically curated dataset with verified DGA family labels, widely used in peer-reviewed research
(3) **360 NetLab DGA Feed** [8]: Industry-sourced malware intelligence from active botnet monitoring

All DGA samples were validated through multiple threat intelligence sources to ensure label accuracy. We acknowledge that dataset quality depends on upstream labeling accuracy, and potential mislabeling could introduce training noise. To mitigate this, we cross-referenced samples across multiple threat feeds and excluded domains with inconsistent labels. The high performance on held-out test data suggests minimal label noise impact.

Responsible Disclosure: We do not release trained model weights publicly to prevent potential adversarial reverse-engineering. However, the methodology is fully described to enable reproducibility by defensive security researchers and practitioners.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors Contribution Statement

Abdhine Ben Ali: Writing original draft, Data curation and processing, Conceptualization. **S. Toklu:** Review, Methodology, Supervision, Validation, Formal analysis. **F. Mugigayi:** Writing, Editing, Visualization, Model Architecture. **M. Djalabi:** Visualization, and Tables.

Acknowledgment

This work has not received any grant or funding.

References

[1] Anderson, H.S., Woodbridge, J., Filar, B., *DeepDGA: adversarially-tuned domain generation and detection*, Proc. ACM Workshop on Artificial Intelligence and Security (AISec), Vienna, Austria, (2016), 13–21.
[2] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S. et al., *From throw-away traffic to bots: detecting the rise of DGA-based malware*, Proc. 21st USENIX Security Symposium, (2012), 491–506.
[3] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T., *Enriching word vectors with subword information*, Trans. Assoc. Comput. Linguist., **5**(2017), 135–146.

[4] Chen, J.L., Qiu, J.F., Chen, Y.H., *A hybrid DGA DefenseNet for detecting DGA domain names based on FastText and deep learning techniques*, IEEE Access, **10**(2022), 42116–42128.
[5] Gregório, J.R., Cansian, A.M., Neves, L.A., Salvadeo, D.H.P., *Deep convolutional neural network and character level embedding for DGA detection*, Proc. 26th Int. Conf. on Enterprise Information Systems (ICEIS), (2024), 167–174.
[6] Mozilla, *cipherscan/top1m*, GitHub. [Online]. Available: https://github.com/mozilla/cipherscan/tree/master/top1m
[7] Namgung, J., Son, S., Moon, Y.S., *Efficient deep learning models for DGA domain detection*, Secur. Commun. Netw., **2021**(2021), 1–18.
[8] 360 Netlab, *DGA feed*. [Online]. Available: http://data.netlab.360.com/dga/
[9] Shahzad, H., Sattar, A.R., Skandaraniyam, J., *DGA domain detection using deep learning*, Proc. 5th IEEE Int. Conf. on Cryptography, Security and Privacy (CSP), Zhuhai, China, (2021), 139–143.
[10] Vaswani, A. et al., *Attention is all you need*, Adv. Neural Inf. Process. Syst. (NeurIPS), (2017), 5998–6008.
[11] Woodbridge, J., Anderson, H.S., Ahuja, A., Grant, D., *Predicting domain generation algorithms with long short-term memory networks*, arXiv:1611.00791, (2016).
[12] Yadav, S., Reddy, A.K.K., Reddy, A.L.N., Ranjan, S., *Detecting algorithmically generated domain-flux attacks with DNS traffic analysis*, IEEE/ACM Trans. Netw., **20**(5)(2012), 1663–1677.
[13] Yu, B., Pan, J., Hu, J., Nascimento, A., De Cock, M., *Character-level based detection of DGA domain names*, Proc. Int. Joint Conf. on Neural Networks (IJCNN), (2018), 1–8.
[14] Zago, M., Gil Pérez, M., Martínez Pérez, G., *UMUDGA - University of Murcia Domain Generation Algorithm Dataset*, Mendeley Data, V1, (2020). [Online]. Available: https://data.mendeley.com/datasets/y8ph45msv8/1