Malicious Domain Names Detection with *DeepDGA*, a Hybrid Character and Word Embeddings Deep Learning Architecture

Lucas Torrealba Aravena*†‡, Pedro Casas†, Diego García*, Javier Bustos-Jiménez*‡, Ivana Bachmann*‡
*NIC Chile Research Labs, †AIT Austrian Institute of Technology,‡ DCC, University of Chile
lucas@niclabs.cl, pedro.casas@ait.ac.at, diegogarcia@niclabs.cl, jbustos@niclabs.cl, ivana@niclabs.cl

***Abstract*—The rapid expansion of the Internet has enabled cybercriminal operations at unprecedented scale. A recurring tactic is the use of *algorithmically generated domains* (AGDs) created by *domain generation algorithms* (DGAs) to orchestrate botnet command-and-control, host phishing content, and distribute malware. Traditional defenses such as blocklists and heuristic rules are brittle against new domains and evolving attacker strategies. We present *DeepDGA*, a hybrid deep learning architecture that fuses character-level and word-level representations to detect both pseudo-random and dictionary-based DGAs. Character-level embeddings processed by a BiLSTM capture subword patterns and entropy; word-level embeddings derived from a dom2words tokenization and Word2Vec capture linguistic regularities exploited by dictionary-based DGAs. Evaluations on a public benchmark with more than 670,000 domains, including 25 DGA families and benign top-popular domains, demonstrate the superiority of *DeepDGA*. The model achieves precision and recall above 0.97 for dictionary-based DGAs, and even higher (above 0.98) for pseudo-random DGAs, consistently outperforming state-of-the-art methods across multiple metrics. *DeepDGA*'s effectiveness, particularly in detecting the more challenging dictionary-based DGAs, highlights the benefit of combining diverse embedding strategies into the same deep learning architecture.**

***Index Terms*—Malicious domains, DGA, AGD, phishing, botnet C&C, deep learning, LSTM, Word2Vec**

I. Introduction

The Internet's rapid expansion has transformed global communication, commerce, and information access, but it has also fueled an escalating wave of cybercrime. Attackers increasingly exploit weaknesses in the DNS system, originally designed as an open and lightweight protocol, to conduct large-scale malicious campaigns. DNS now serves as a resilient backbone for botnets, phishing, ransomware, and malware distribution, resulting in billions of dollars in losses worldwide. Central to these campaigns is the use of *Algorithmically Generated Domains* (AGDs), which malware generates in bulk through *Domain Generation Algorithms* (DGAs). Infected machines cycle through these domains until one connects to an attacker-controlled Command and Control (C&C) server. This creates a severe asymmetry: while adversaries need only register a few domains, defenders must monitor and block vast, constantly changing sets. A notorious case is Conficker.C, which generated 50,000 domains daily but required only a small subset for successful communication.

Beyond C&C operations, DGAs underpin other forms of abuse. AGDs host phishing pages to steal sensitive information, act as disposable infrastructure for spam, and enable malware propagation while evading blacklist-based defenses. DGAs can be broadly categorized as random or dictionary-based. Random DGAs produce gibberish-like strings (e.g., "jwdumixczs.com") that are often detectable with statistical heuristics, whereas dictionary-based DGAs concatenate real words into plausible names (e.g., "translateincoming.com"). The latter are particularly deceptive, as they closely resemble legitimate domains and thus remain difficult to detect with traditional rule-based or character-level approaches, posing one of the most persistent challenges in DGA detection.

Researchers have explored multiple strategies for DGA detection, ranging from blocklists and heuristic methods to machine learning and deep learning approaches. While blocklists and handcrafted features offer limited adaptability, deep learning models such as CNNs and LSTMs have achieved strong results by learning directly from raw domain strings. Nevertheless, reliably detecting dictionary-based DGAs remains an open challenge.

In this work, we present *DeepDGA*, a hybrid deep learning framework that integrates two complementary perspectives on domain names. A character-level encoder based on a Bidirectional LSTM captures sequential and morphological cues typical of pseudo-random DGAs, such as digit-letter alternations and entropy spikes. A word-level encoder leverages *dom2words* tokenization and pre-trained Word2Vec embeddings to introduce semantic awareness, enabling effective discrimination of domains constructed from real words or dictionary-derived tokens. By fusing these representations, *DeepDGA* achieves robust classification across both random and dictionary-based DGAs. Our study relies solely on domain strings, avoiding DNS records or traffic data. This choice ensures scalability, preserves privacy, and enables lightweight deployment at registrars, resolvers, or gateways, including pre-registration screening.

This paper makes three key contributions. First, it introduces *DeepDGA*, a novel hybrid architecture that combines character-level embeddings via a Bidirectional LSTM with word-level embeddings derived from domain segmentation and a pre-trained Word2Vec model. This dual design effectively addresses both pseudo-random DGAs and the more challenging dictionary-based DGAs that mimic legitimate domains. Second, extensive experiments show that *DeepDGA* significantly outperforms existing approaches, achieving over 0.97 precision and recall on dictionary-based DGAs, and maintaining a TPR above 95% with a FPR below 2% – a

978-3-903176-75-1 ©2025 IFIP

benchmark unmatched by other evaluated models. Finally, the paper provides a comprehensive benchmarking study against state-of-the-art machine learning and deep learning methods. The evaluation covers diverse scenarios, including severe class imbalance and comparisons across dictionary-based and pseudo-random DGAs to demonstrate the robustness and generalization of *DeepDGA*.

The findings demonstrate that *DeepDGA* delivers strong recall and ROC-AUC even under severe imbalance and significantly reduces the performance gap observed on dictionary-based families when compared to character-only models. These results suggest that combining character-level and word-level embeddings constitutes a practical and effective path forward for AGD detection. Beyond its accuracy, *DeepDGA* offers a deployable solution: it can act as a lightweight, privacy-preserving filter, augment blocklists with predictive coverage of unseen domains, and provide an analytic foundation for real-time defenses against rapidly evolving DGA threats.

II. Related Work

DGA detection, particularly through domain name analysis, has seen extensive research across various methodologies: Rule-Based or Heuristic Analysis, Machine Learning, and Deep Learning. Each approach contributes distinctively to defense strategies, yet each also presents unique advantages and limitations.

a) Rule-Based or Heuristic Analysis: this method identifies AGDs by deploying predefined logic or criteria, often based on specific characteristics of AGDs. A significant advantage is the interpretability it offers, enhancing understanding of system functionalities. However, its static nature makes it vulnerable to behavioral changes by attackers, limiting its resilience. Blocklists, which identify domains already known to be malicious, are a common rule-based approach, but their predictive capability is restricted to the frequency of updates, offering limited defense against newly generated domains [1], [2]. More sophisticated rule-based methods include statistical approaches like those based on n-grams, as proposed by Zhao et al. [3], which calculate a reputation value for domains. While achieving a detection accuracy of 94.04%, this method suffers from a manual threshold selection and relatively high false positive and negative rates. Sharifnya et al. [4] introduced DFBotKiller, which combines statistical measures and a negative reputation system with DNS failure history to detect domain-flux botnets.

b) Machine Learning: machine learning techniques analyze human-defined features (e.g., entropy, length, n-grams) extracted from AGDs using traditional classifiers or ensemble methods. These approaches leverage statistical models to recognize malicious patterns, offering adaptability to evolving threats and improved detection accuracy over time. However, they can be complex and resource-intensive to train and maintain. Several notable works have advanced DGA detection through machine learning. Bilge et al. proposed Exposure [5], which uses a J48 decision tree classifier with 15 features based on time, DNS answer, and domain characteristics, achieving 99.5% detection with false positives below 1%. Wang et al. [6] applied machine learning algorithms on distance metrics such as Kullback-Leibler Distance, Edit Distance, and Jaccard Index, achieving over 99% accuracy for tested AGDs. Cucchiarelli et al. [7] employed lexical features (2-grams and 3-grams) for similarity expression, showing high accuracy in classifying DGA-based domains. Da Luz [8] extracted 36 features from passive DNS data, including lexical (e.g., character diversity, n-gram statistics) and network-related characteristics, demonstrating that Random Forest yielded the best results with 97% accuracy and a 3% False Positive Rate. Selvi et al. [9] enhanced DGA detection using Random Forest by combining Da Luz's features with 'masked n-grams," where domain names are transformed (vowels to v', consonants to c', numbers to n') to calculate substring occurrences. Antonakakis et al. [10] proposed Pleiades, which focuses on detecting NXDOMAIN responses, grouping them with DGA-generated and legitimate domains, and using statistical features (name length, randomness, n-gram distribution frequency) with a Hidden Markov Model.

c) Deep Learning: a subset of machine learning, deep learning utilizes neural networks to discern intricate patterns within AGDs, learning from extensive datasets. This approach offers high adaptability to evolving botnet tactics and continuous improvement in detection accuracy. A primary challenge, however, is the limited interpretability of deep learning models, making it difficult to understand the reasoning behind specific predictions. Key developments include: Long Short-Term Memory (LSTM) Networks have shown highly effective for AGD detection, due to their ability to capture long-short term dependencies in temporal sequences. Woodbridge et al. [11] applied an LSTM network to raw URL character sequences, achieving high effectiveness (0.9993 AUC, 0.9906 F1 score). by Tran et al. proposed LSTM.MI [12], an advanced LSTM algorithm addressing the multiclass imbalance problem prevalent in DGA malware detection, outperforming original LSTMs and other cost-sensitive methods, showing significant improvements in recall and precision and serving as a key baseline. Using Convolutional Neural Networks (CNNs), Catania et al. [13] proposed a neural network combining an embedding layer, a 1D-CNN, and a dense layer, achieving a TPR of approximately 97% with a FPR of 0.7%. Recurrent CNNs were also applied; Liu et al. [14] proposed RCNN-SPP, combining a Bidirectional LSTM (Bi-LSTM) layer with convolutional layers and a modified pooling algorithm, achieving 92% accuracy and 90% F1-score. Finally, Yang et al. [15] introduced an architecture based on heterogeneous Deep Learning, using parallel convolutional layers for local features (IPCNN) and a Self-Attention based Bi-LSTM (Sa-Bi-LSTM) for global features, demonstrating superior performance.

III. DeepDGA: A Hybrid AGD Detector

The proposed model, *DeepDGA*, is a hybrid architecture that concatenates two complementary approaches – character-level embeddings and word-level embeddings – for robust

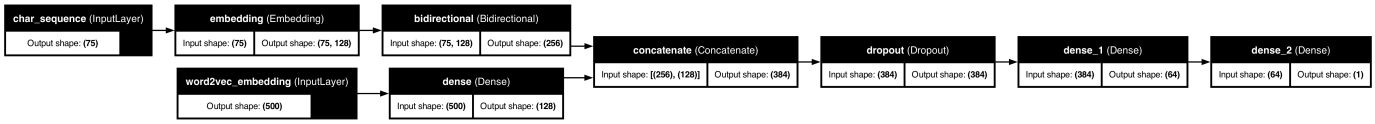


Fig. 1: Proposed *DeepDGA* architecture implemented in TensorFlow.

detection of algorithmically generated domains. This design explicitly addresses the dual challenges of detecting random or pseudo-random DGAs and dictionary-based DGAs, the latter being particularly difficult to detect due to their similarity to legitimate domain names. As illustrated in Figure 1, *DeepDGA* comprises two independent encoders whose outputs are fused into a joint latent representation for classification. The intuition behind this architecture is straightforward: character-level features excel at capturing structural irregularities and entropy-based patterns typical of pseudo-random DGAs, whereas word-level embeddings, learned through natural language processing (NLP) techniques, provide semantic cues that improve detection of dictionary-based DGAs. By combining these two views, *DeepDGA* achieves balanced robustness across DGA families.

A. Character-Level Encoder

The first encoder operates at the character level, mapping each domain string into a sequence of character embeddings. Each domain is normalized to a fixed length of $L = 75$ characters, following the methodology of Woodbridge et al. [11]. This choice ensures that all domains fit into a common input size, using padding where necessary without information loss.

Each character is embedded into a 128-dimensional vector space via an `embedding` layer, producing a matrix of size 75×128. This sequence is then processed by a Bidirectional Long Short-Term Memory network, with 128 hidden units per direction, following the architecture introduced by Mac et al. [17]. The BiLSTM captures dependencies in both forward and backward directions, enabling it to model lexical and structural properties such as alternating letters and digits, irregular sub-sequences, and local entropy spikes. The final hidden states from both directions are concatenated, yielding a 256-dimensional latent representation of the domain at the character level $z_d^{char} \in \mathbb{R}^{256}$. This representation is particularly effective for random or pseudo-random DGAs, where structural anomalies serve as strong discriminators.

B. Word-Level Encoder

Dictionary-based DGAs generate domains that resemble legitimate names by concatenating meaningful words or tokens. To capture this semantic information, *DeepDGA* employs a word-level encoder based on Word2Vec, as previously introduced by Torrealba et al. [20].

Tokenization (dom2words): Each domain d is segmented into a sequence of tokens using frequency-aware segmentation, referred to as *dom2words*. For example, the dictionary-based domain `mortiscontrastatim.com` is segmented into:

$$s_d = \{\texttt{mortis}, \texttt{contrast}, \texttt{a}, \texttt{tim}, \texttt{com}\}$$

Word Embeddings: Each token $w_j \in s_d$ is mapped to an embedding z_{w_j} using a pre-trained Word2Vec model trained on a large web corpus. We employ the skip-gram architecture, which predicts surrounding words given a center word, weighting nearby words more heavily than distant ones. The embedding dimensionality is fixed at $\gamma = 100$, and a context window of length $l = 5$ is used.

Aggregation: To obtain a domain-level representation from the individual token embeddings, five aggregation functions are computed: z_d^{min}, z_d^{mean}, z_d^{max}, z_d^{sum}, $z_d^{TF\text{-}IDF}$. Here, the first three correspond to element-wise pooling (minimum, mean, maximum), z_d^{sum} is the element-wise sum, and $z_d^{TF\text{-}IDF}$ is a weighted sum using Term Frequency-Inverse Document Frequency (TF-IDF). These five aggregated vectors are concatenated to form a 500-dimensional domain representation $z_d^{word} \in \mathbb{R}^{500}$.

A dense layer with 128 ReLU units is then applied to refine and reduce this representation:

$$\tilde{z}_d^{word} = f_{\text{ReLU}}(W z_d^{word} + b), \quad \tilde{z}_d^{word} \in \mathbb{R}^{128}$$

This word-level embedding strategy is especially important for detecting dictionary-based DGAs, as it captures semantic information that purely character-level approaches cannot.

C. Fusion and Final Classification

The outputs of the two encoders are concatenated to form a unified 384-dimensional latent vector:

$$z_d = [z_d^{char} \,\|\, \tilde{z}_d^{word}] \in \mathbb{R}^{384}$$

To improve generalization, dropout with rate $p = 0.5$ is applied, followed by a fully connected layer of 64 units with ReLU activation, enabling the model to learn non-linear feature interactions. The final classification is performed by a single sigmoid neuron, outputting $\hat{y} \in [0, 1]$ as the probability that the domain is malicious:

$$\hat{y} = \sigma(W z_d + b)$$

The model is trained with the binary cross-entropy loss function:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \big[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \big],$$

where $y_i \in \{0, 1\}$ is the true label, and optimized using Adam for efficient convergence.

Overall, *DeepDGA* demonstrates that combining sequential morphology from characters with semantic patterns from words yields a balanced and effective detector. As we show next, this architecture enables robust detection across a diverse range of DGA families, particularly closing the gap

on dictionary-based domains where traditional approaches underperform.

IV. Experimental Evaluations and Results

To assess the performance of *DeepDGA*, we conducted extensive experiments and comparisons with the most relevant models in the literature. We replicated baseline methods that have demonstrated strong results in prior studies, ensuring that all models were trained and evaluated under the same dataset and metrics, and that deep learning models were trained for the same number of epochs to ensure a fair comparison.

The benchmark includes both classical machine learning and deep learning approaches. On the machine learning side, we consider works such as Bilge et al. [5], Da Luz [8], Almashhadani et al. [16], Cucchiarelli et al. [7], and Selvi et al. [9], which rely on handcrafted or statistical features extracted from domain names or DNS data, sometimes extended with masked n-gram representations.

On the deep learning side, we replicate several representative models that learn directly from raw or tokenized domain strings. These include the LSTM-based approaches of Woodbridge et al. [11] and Tran et al. [12], the Bidirectional LSTM of Mac et al. [17], the deep learning variant proposed by Selvi et al. [18], and the more recent hybrid CNN-LSTM architecture of Zang et al. [19].

In the presented results next, and in particular in the different figures, *DeepDGA* is referred to as the model by Torrealba.

A. Dataset

We use a public, balanced dataset containing 674,898 domains: benign samples drawn from the Alexa Top Sites and malicious samples uniformly distributed across 25 DGA families.[1] Following prior work [7], DGA families are grouped as (i) **dictionary-based**, including 4 families that concatenate real words to generate readable domains (*gozi*, *nymaim*, *matsnu*, *suppobox*) and (ii) **pseudo-random**, including 21 families that generate high-entropy, unreadable domains (*conficker*, *vawtrak*, *simda*, etc.). Although Alexa's ranking service was discontinued in 2022, its dataset remains a widely adopted benchmark for benign domains.

B. Experimental Setup

Two experimental scenarios were designed, which we refer to as Experiment 1 and Experiment 2.

Experiment 1 – Class Balance and Imbalance: here we study robustness under three splits of malicious vs. benign domains: (90%,10%), (50%,50%), and (10%,90%). This simulates prevalence shifts between training and deployment. For each scenario, models are trained on one distribution and evaluated across all three. **Case 1 (90/10)**: represents settings where synthetic malicious data dominates training. An example of this case is when access to the DGA algorithm is available, and large scale variations of synthetic data are created to train the model for specific DGA identification.

[1]Dataset available at: https://github.com/chrmor/DGA_domains_dataset

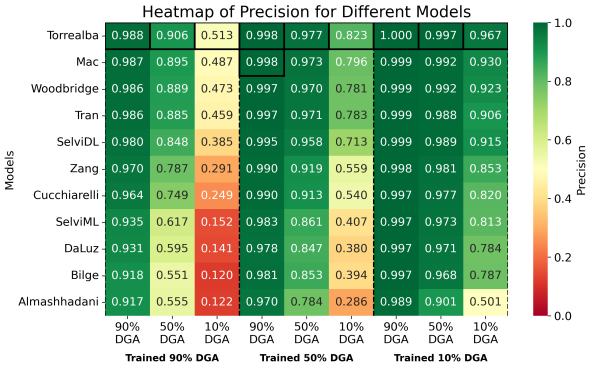


Fig. 2: Precision (Experiment 1: Class Balance/Imbalance).

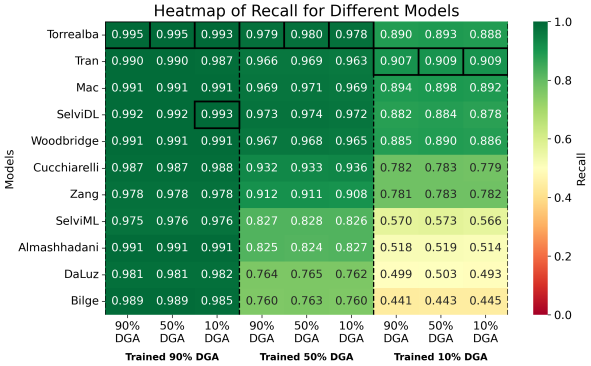


Fig. 3: Recall (Experiment 1: Class Balance/Imbalance).

Case 2 (50/50): a balanced benchmark, standard in research but less representative of real-world deployment. **Case 3 (10/90)**: represents scenarios where DGA domain names have been identified, and the model is trained with these domains and evaluated in a real environment.

Experiment 2 – Dictionary-based vs. Pseudo-random: we build two balanced datasets, each composed of 50% benign and 50% malicious domains. One uses only the four dictionary-based families, and the other uses the 21 pseudo-random families. Both share the same benign pool. The objective is to compare performance across different generation strategies, highlighting the particular difficulty of dictionary-based DGAs due to their human-like readability.

C. Results Experiment 1: Class Balance/Imbalance

Figures 2–5 report the results of Experiment 1 across all benchmarked models, in terms of precision, recall, F1-score, and ROC–AUC. The proposed *DeepDGA* (labeled as Torrealba) consistently achieves the best or near-best performance in every metric, even under severe class imbalance.

In terms of ROC-AUC, *DeepDGA* maintains values above 0.995 across all regimes, clearly outperforming traditional machine learning approaches, which experience significant degradation under skewed distributions. Precision results highlight the expected drop in the 10% malicious case due to low prevalence, yet *DeepDGA* still outperforms competing

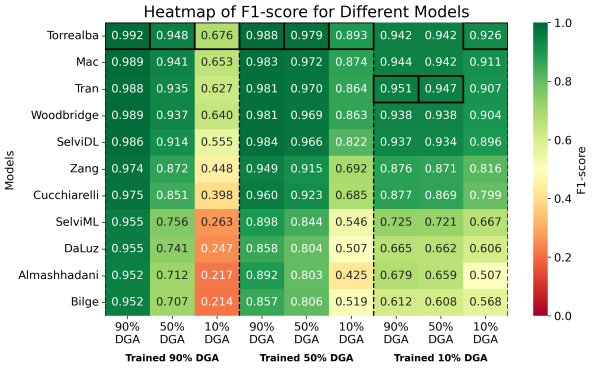


Fig. 4: F1-score (Experiment 1: Class Balance/Imbalance).

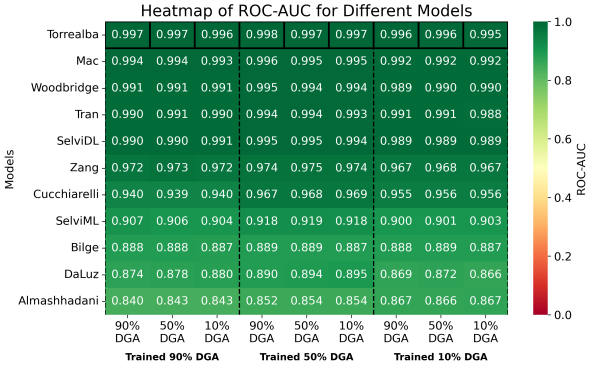


Fig. 5: ROC-AUC (Experiment 1: Class Balance/Imbalance).

methods, most of which collapse below 0.5. Recall remains robust, with *DeepDGA* sustaining values above 0.88 across all imbalance settings, confirming its ability to detect minority-class signals even when positive instances are rare.

The F1-scores further emphasize the model's resilience, staying above 0.67 in the most imbalanced case (10% malicious) while other models degrade sharply, in some cases dropping below 0.3. Overall, these findings confirm that the hybrid embedding architecture of *DeepDGA* provides strong robustness against class imbalance, significantly outperforming models based on handcrafted features or single-stream embeddings.

D. Results Experiment 2: Dictionary vs. Pseudo-random

Figures 6–7 compare models across dictionary-based and pseudo-random subsets, in terms of precision and accuracy. *DeepDGA* outperforms all baselines on dictionary-based DGAs, achieving precision above 0.97, and matches the top result above 0.98 on pseudo-random families. Classical machine learning models (Cucchiarelli, SelviML, Da Luz, Bilge, Almashhadani), which rely on features such as character diversity, digit counts, or substring length, are struggling on dictionary-based domains, with precision below 0.89. This limitation arises because dictionary-based domains strongly resemble benign domains.

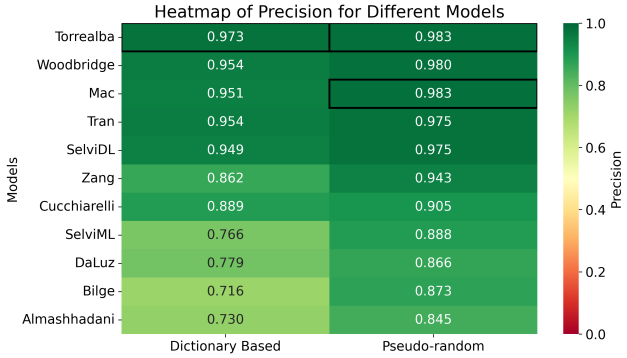


Fig. 6: Precision (Experiment 2: Dict. vs. Pseudo-random).

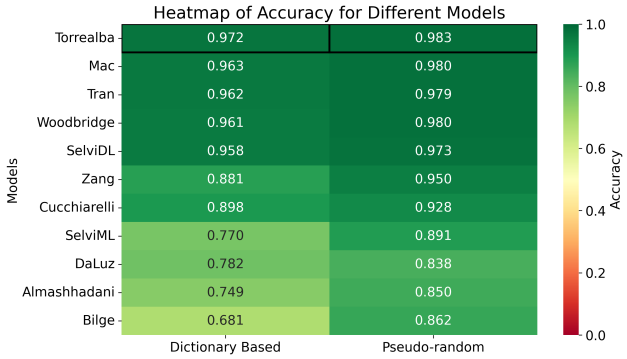


Fig. 7: Accuracy (Experiment 2: Dict. vs. Pseudo-random).

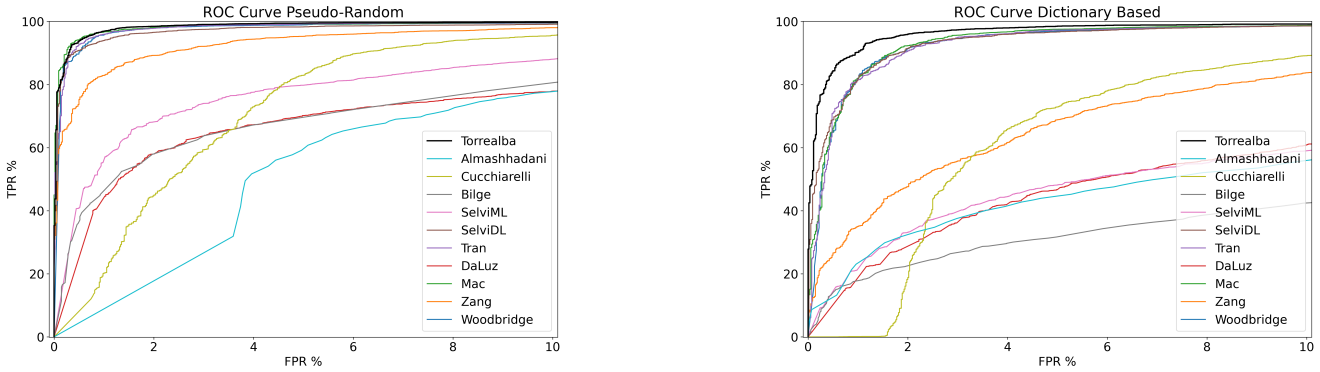
Zang's hybrid CNN-LSTM model achieves solid performance on pseudo-random families (94% precision) but fails to exceed 87% on dictionary-based families. In contrast, deep learning approaches leveraging embeddings (Woodbridge, Mac, SelviDL, Tran, and *DeepDGA*) capture richer patterns and reach precision above 0.94 on dictionary-based DGAs and above 0.97 on pseudo-random ones.

Accuracy results confirm the same trend in Figure 7. *DeepDGA* in particular achieves the best overall scores, exceeding 0.97 for dictionary-based and 0.98 for pseudo-random DGAs. Machine learning baselines peak at 0.90 accuracy for dictionary-based families, while deep learning models range from 0.88 to 0.97.

To conclude, Figure 8 presents the ROC curves for Experiment 2, for both (a) pseudo-random and (b) dictionary-based DGAs. *DeepDGA* shows clear strength across cases: for dictionary-based DGAs it achieves a TPR above 95% while maintaining FPR below 2%, and for pseudo-random DGAs it achieves TPR above 97% with FPR below 1%. No other benchmarked model is able to simultaneously sustain such high detection rates and such low false alarm rates, underscoring the robustness of the proposed hybrid architecture in both challenging scenarios.

V. Concluding Remarks

We introduced *DeepDGA*, a hybrid character- and word-level embedding model for AGD detection. Experiments on



(a) ROC-AUC Experiment 2: Pseudo-Random

(b) ROC-AUC Experiment 2: Dictionary-based

Fig. 8: ROC curves for Experiment 2. *DeepDGA* consistently achieves the best trade-off between sensitivity (TPR) and specificity, outperforming all baselines.

a large public dataset demonstrate strong and balanced performance across class imbalance regimes and DGA types, with particularly notable improvements in the detection of dictionary-based DGAs. Future work will extend this architecture with attention mechanisms to improve interpretability and will investigate its integration into large-scale inference systems, including latency and memory profiling.

The results validate the design choices underlying *DeepDGA*. Its hybrid architecture consistently outperforms models that rely solely on lexical statistics or single-stream embeddings. Notably, *DeepDGA* closes the long-standing performance gap on dictionary-based DGAs while maintaining excellent results on pseudo-random families. It also achieves high ROC–AUC under severe class imbalance, demonstrating resilience to prevalence shifts. Moreover, by relying only on the domain string, the model supports passive and privacy-preserving deployment, making it suitable for integration into registrars, resolvers, and enterprise gateways at scale.

Despite these advantages, certain limitations remain. Interpretability is lower than in rule-based methods, and incorporating attention mechanisms could help increase analyst trust. The accuracy of word-level segmentation (*dom2words*) may also introduce errors for some domain families, and the pre-trained Word2Vec embeddings require periodic updates to remain effective. Nevertheless, the experimental evidence strongly supports *DeepDGA* as a robust, efficient, and practical solution for detecting algorithmically generated domains in real-world settings.

Acknowledgment

This work has been supported by the FWF Austrian Science Fund, Project reference I-6653, as part of the EU CHIST-ERA-2022-SPiDDS-02 project *GRAPHS4SEC* and by ANID - Agencia Nacional de Investigación y Desarrollo, Chile, through the Doctorado Nacional scholarship, under Grant No. 1808/2023.

References

[1] M. Botacin, F. Ceschin *et al.*, "Challenges and Pitfalls in Malware Research," *Computers & Security*, vol. 106, p. 102287, 2021.

[2] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, and M. K. Khan, "Ransomware: Recent Advances, Analysis, Challenges and Future Research Directions," *Computers & security*, vol. 111, p. 102490, 2021.

[3] H. Zhao, Z. Chang, G. Bao, X. Zeng *et al.*, "Malicious Domain Names Detection Algorithm based on n-gram," *Journal of Computer Networks and Communications*, vol. 2019, 2019.

[4] R. Sharifnya and M. Abadi, "Dfbotkiller: Domain-flux botnet detection based on the history of group activities and failures in dns traffic," *Digital Investigation*, vol. 12, pp. 15–26, 2015.

[5] L. Bilge *et al.*, "EXPOSURE: Finding Malicious Domains using Passive DNS Analysis," in *Proc. of the NDSS Symposium*, USA, 2011.

[6] Z. Wang, Y. Guo, and D. Montgomery, "Machine learning-based algorithmically generated domain detection," *Computers and Electrical Engineering*, vol. 100, p. 107841, 2022.

[7] A. Cucchiarelli, C. Morbidoni, L. Spalazzi, and M. Baldi, "Algorithmically generated malicious domain names detection based on n-grams features," *Expert Systems with Applications*, vol. 170, p. 114551, 2021.

[8] P. M. Da Luz, "Botnet detection using passive dns," *Radboud University: Nijmegen, The Netherlands*, vol. 25, p. 27, 2014.

[9] J. Selvi, R. J. Rodriguez, and E. Soria-Olivas, "Detection of Algorithmically Generated Malicious Domain Names using Masked n-grams," *Expert Systems with Applications*, vol. 124, pp. 156–163, 2019.

[10] M. Antonakakis *et al.*, "From {Throw-Away} Traffic to Bots: Detecting the Rise of {DGA-Based} Malware," in *21st USENIX Security Symposium (USENIX Security 12)*, 2012, pp. 491–506.

[11] J. Woodbridge, H. S. Anderson, A. Ahuja, and D. Grant, "Predicting domain generation algorithms with long short-term memory networks," *arXiv preprint arXiv:1611.00791*, 2016.

[12] D. Tran, H. Mac, V. Tong, H. A. Tran, and L. G. Nguyen, "A lstm based framework for handling multiclass imbalance in dga botnet detection," *Neurocomputing*, vol. 275, pp. 2401–2413, 2018.

[13] C. Catania, S. García, and P. Torres, "Deep convolutional neural networks for dga detection," in *Computer Science–CACIC 2018: 24th Argentine Congress*, Revised Selected Papers, Springer, 2019.

[14] Z. Liu *et al.*, "Detection of Algorithmically Generated Domain Names using the Recurrent Convolutional Neural Network with Spatial Pyramid Pooling," *Entropy*, vol. 22, no. 9, p. 1058, 2020.

[15] L. Yang, G. Liu, Y. Dai, J. Wang, and J. Zhai, "Detecting stealthy domain generation algorithms using heterogeneous deep neural network framework," *IEEE Access*, vol. 8, pp. 82 876–82 889, 2020.

[16] A. O. Almashhadani, M. Kaiiali, D. Carlin, and S. Sezer, "Maldomdetector: A system for detecting algorithmically generated domain names with machine learning," *Computers & Security*, vol. 93, p. 101787, 2020.

[17] H. Mac *et al.*, "DGA Botnet Detection using Supervised Learning Methods," in *Proceedings of the 8th International Symposium on Information and Communication Technology*, 2017, pp. 211–218.

[18] J. Selvi, R. J. Rodríguez, and E. Soria-Olivas, "Toward optimal lstm neural networks for detecting algorithmically generated domain names," *IEEE Access*, vol. 9, pp. 126 446–126 456, 2021.

[19] X. Zang, J. Cao, X. Zhang, J. Gong, and G. Li, "Botdetector: a system for identifying dga-based botnet with CNN-LSTM," *Telecommun. Syst.*, vol. 85, no. 2, pp. 207–223, 2024.

[20] L. Torrealba Aravena, P. Casas, J. Bustos-Jiménez, G. Capdehourat, and M. Findrik, "Dom2Vec: Detecting DGA Domains Through Word Embeddings and AI/ML-Driven Lexicographic Analysis," in *Proc. 19th International Conference on Network and Service Management (CNSM)*, 2023, pp. 1–5.