Dom2Vec - Detecting DGA Domains through Word Embeddings and AI/ML-driven Lexicographic Analysis

L. Torrealba Aravena*†, P. Casas*, J. Bustos-Jiménez†, G. Capdehourat‡, M. Findrik§
†NIC Labs, Universidad de Chile, *AIT Austrian Institute of Technology
‡Universidad de la República & Plan Ceibal, §cyan Security Group

***Abstract*—The timely identification of DNS queries to Domain Generation Algorithm (DGA) domains plays a critical role in mitigating malware propagation and its potential impact, especially in thwarting coordinated botnet activity. We introduce *Dom2Vec*, an innovative approach for swiftly detecting DGA-generated domains by leveraging lexicographic features exclusively derived from the observed domain names in DNS queries. *Dom2Vec* leverages word embeddings to map tokens extracted from domain names into highly expressive representations. These representations are then combined with a reputation-based scoring system for domain names, which utilizes the co-occurrence frequency of n-grams in relation to a list of whitelisted domains. The fusion of domain embeddings, reputation scores, and other meaningful lexicographic features derived from domain names provides robust domain name representations for AI/ML-driven detection of DGAs. Through experimental evaluation on a dataset comprising 25 distinct families of DGA domains, we demonstrate that *Dom2Vec* significantly outperforms current state-of-the-art approaches for DGA detection and analysis, improving our previous detection system based on reputation scores by at least 30%, for a false-alarm rate below 1%.**

***Index Terms*—DGA Detection, Word2Vec, TF-IDF, n-grams, Lexicographic Analysis, DNS, Machine Learning.**

I. Introduction

Botnets heavily rely on C&C servers to coordinate bots, i.e., compromised machines. To evade detection, botnets often employ Domain Generation Algorithms (DGAs) that generate a diverse set of (quasi) random domain names based on a seed parameter, sometimes relying on pre-defined dictionaries [1]. Detecting and neutralizing the C&C server domain name is therefore a key strategy to combat botnets. While DGA domains are primarily associated with malware and botnets, they can also be utilized for phishing purposes. Attackers may register DGA-generated domains that closely resemble popular brands or services, intending to deceive victims into thinking they are interacting with trusted entities.

The most common techniques to detect malicious websites is to rely on filtering blocklists – filtering here corresponds to exact matching, Levenshtein distance [2], etc. While blocklisting is efficient and simple to implement and interpret, it also presents a series of limitations on its usage [3]. Therefore, this strategy is generally adopted as a first filter to protect users from well-known phishing attacks. Multiple approaches in the state of the art rely on domain associated features as input to detect malicious domains. Detection is typically done using hard-coded rules, (shallow) machine learning derived rules, or more complex, representation-learning and deep-learning based functions. Among the most relevant features considered in the literature, we identify *lexicographic features* – features derived from the lexicographic analysis of the domain name itself [3], and *content-based features* – features based on the content of the website associated to the domain (e.g., full HTML contents).

We propose an approach to detection of DGA-generated domains by analyzing lexicographic features derived exclusively from the domain name, as observed in the monitored DNS queries. This approach allows for efficient computation and large-scale monitoring, as the features are derived solely from processing the domain name, eliminating the need for external information sources. Moreover, by not accessing the content of a domain itself, privacy preservation for end-users is significantly enhanced.

Lexicographic analysis, and in particular n-grams, has been largely explored for classification of domain names. In the realm of machine learning for domain name analysis, various approaches have been investigated [4], [5], [6], [7], including Random Forest models, XGBoost classifiers, and CNNs. These models leverage a combination of lexicographic-based, content-based, and other relevant features to achieve high accuracy in identifying malicious domains. The usage of n-grams in self-explainable approaches, without relying on learning models, is also part of the state of the art [2], [8].

A. Domain Reputation Scores

Following state of the art [8], our method employs a reputation-based scoring system for domain names, utilizing the co-occurrence frequency of n-grams as compared to a whitelist of well-known benign domains. To identify DGA domains, we segment the domain name into n-grams of varying lengths and use them to calculate a reputation or similarity score against the list of n-grams obtained from the whitelisted domains. The resulting reputation score is finally compared against a predefined detection threshold for DGA/non-DGA binary classification. Additionally, we extract meaningful lexicographic features from domain names and leverage machine learning techniques to enhance the effective-

978-3-903176-59-1 ©2023 IFIP

ness of detection. Lexicographic features encompass various characteristics, including domain length, the randomness of the characters, and the frequency of n-grams, the latter by using the computed reputation scores.

B. Dom2Vec Domain Embeddings

While this initial set of features can properly differentiate between DGA and non-DGA domains, they tend to fail for DGA approaches based on dictionary words. Dictionary-driven DGAs generate domain names that appear more similar to legitimate domains, making it harder to differentiate them. We therefore explore a more expressive approach to represent the lexicographic content of domain names, in particular by leveraging Word2Vec embeddings [9], [10]. Word2Vec is a Natural Language Processing (NLP) technique based on neural networks which maps words or *tokens* of text sentences into a latent space as a real-valued vector – the *embedding*, such that words belonging to similar contexts have similar embeddings. The core element of the Word2Vec model is the context, which is defined as the sequence of words surrounding the specific word for which the embedding is computed. To generate such context or sentence out of a domain name, we resort to NLP techniques for *text splitting*, based on the frequency of words as observed in a predefined large-size *learning corpora* of documents. In a nutshell, given a domain name, we split it in a set of known-words (from the learning corpora) if these are present in the domain name, or in a set of tokens when no words are identified. For example, the domain name $d =$ `mortiscontrastatim.com` (generated by the dictionary-based DGA `gozi`) is transformed in the sentence $s_d =$ {'mortis', 'contrast', 'a', 'tim', 'com'}, whereas the domain name $d =$ `cvyh1po636avyrsxebwbkn7.ddns.net` (generated by the DGA `corebot`) is transformed into $s_d =$ {'c', 'vy', 'h', '1', 'po', '636', 'av', 'yrs', 'x', 'eb', 'wb', 'kn', '7', 'd', 'dns', 'net'}. Word2Vec is then applied on top of the resulting sentences s_{d_i}, obtaining as such an embedding for each word observed in a training dataset. Finally, an embedding for domain d is computed out of the embeddings of each of the words in s_d, using different pooling techniques, such as min, max, average, etc. Identification of known words in a domain name additionally helps to counteract the negative impact of dictionary-based DGAs on detection performance, as the resulting embeddings can better capture the underlying dictionaries and patterns behind such DGAs. We refer to the combination of the domain embeddings with the aforementioned reputation scores and lexicographic features as the *Dom2Vec* representation of a domain name. Using *Dom2Vec* representations, we train supervised learning models for binary detection of DGAs.

Word2Vec has been used in previous work to model domain names [11], [12], but in a completely different manner as we apply it in *Dom2Vec*. In particular, [11] uses Word2Vec to capture the interactions between end hosts and domains, characterizing time-series patterns of DNS queries for each IP address, and [12] uses it to learn the semantic of subsequent domain names from users' web navigation patterns.

We evaluate the performance of the different proposals on a publicly available dataset of domains names, consisting of a list of well-known domains (considered as benign) and a list of DGA-generated domains, using 25 different DGA families [1]. Results show that: (i) n-gram-based reputation scores can discriminate between DGA and non-DGA domains for different DGA algorithms; (ii) while the proposed reputation score significantly improves detection over state of the art [8], if fails to properly detect DGAs when these are generated through dictionary-based approaches; (iii) detection performance can be highly improved by using even simple learning models, combining reputation scores with other lexicographic features and domain embeddings. This paper builds upon *PHISHWEB*, our recent work on web phishing detection [13], with a particular focus on detection of DGAs and embedding of domain names as novel contributions.

II. DGA Detection with RVP and RF3

The initial DGA detector we introduce is a variation of a previously proposed algorithm for detection of malicious domains [8], which is based on the computation of a *reputation score* or value RV for the domain under analysis. In a nutshell, a domain name is segmented in n-grams of different length n, and a score is computed based on the occurrence of these n-grams on a set of well-known benign names, serving as a reputation list. We refer to our version of the reputation score as Reputation Value *PHISHWEB* or RVP. For a certain domain d, RVP is defined as follows:

$$\text{RVP}(d) = \sum_{i=1}^{m} \frac{n}{\lambda_d} \times W_n(i) = \sum_{i=1}^{m} \frac{n}{\lambda_d} \times log_2 \left(\frac{N_n(i)}{n} \right)$$

where m is the total number of n-grams derived from d, for $n = 3$ to 7, λ_d is the length of d, and $N_n(i)$ is the total number of occurrences of n-gram i in the reputation list of domains. The idea of the reputation value is to reflect how similar are the sub-strings of domain d to the list of *benign* sub-strings in this list. The higher the value of RVP, the higher the chances of d being a benign domain. Detection is achieved by simple thresholding on RVP. As we show in the results, RVP drastically improves detection performance over the former score RV [8], which is strongly biased by the length of a domain, limiting its usefulness in the practice.

While RVP provides highly accurate DGA detection performance, results can be further boosted by combining RVP with a small set of simple lexicographic features, through a machine learning driven approach. In particular, we consider a 3-tuple to characterize a domain d, including its RVP score RVP(d), its length λ_d, and a measure of the randomness of its characters $H(d)$, and use it as input to a standard random forest (RF) model, trained for binary classification. $H(\cdot)$ corresponds to the empirical entropy of the characters composing the domain name. We refer to this detector as RF3.

III. *Dom2Vec* - Embeddings for Domain Names

Dom2Vec leverages the power of Word2Vec to generate embeddings for the sequences of words (domain sentences)

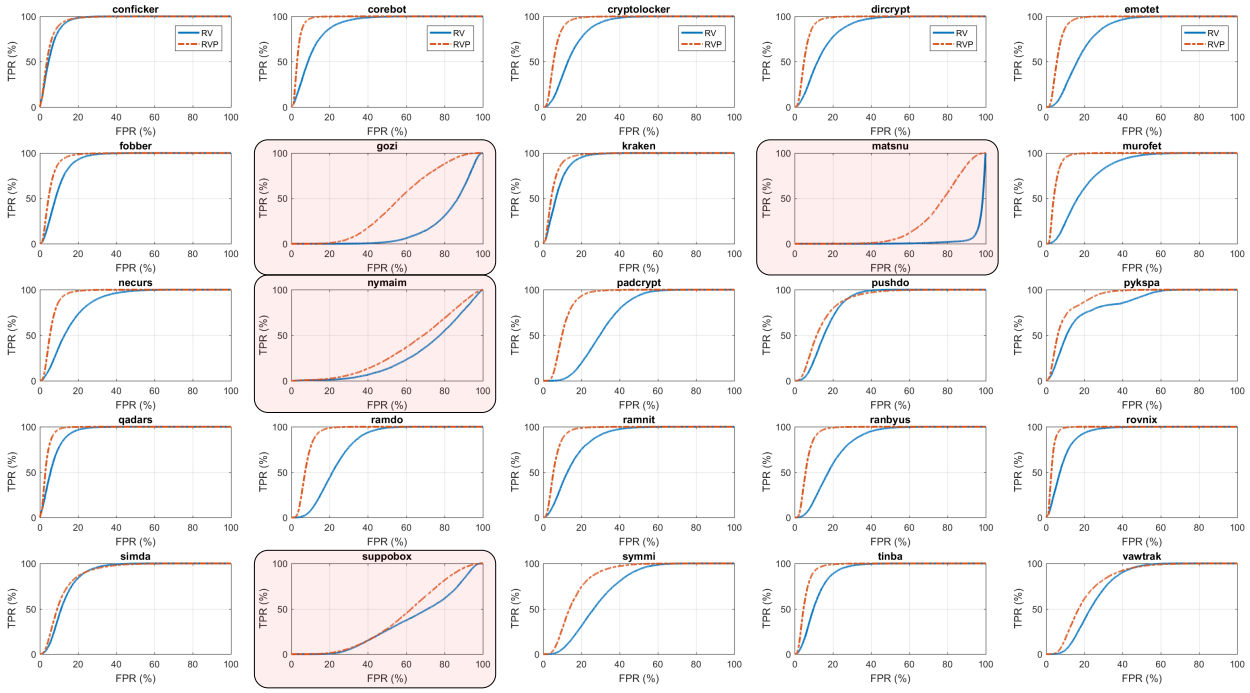


Figure 1. Individual DGA-algorithm detection performance for RV and RVP scores, reported as ROC curves. RVP clearly outperforms RV for all DGA families, but fails to properly detect dictionary-based DGAs: `gozi`, `suppobox`, `nymaim`, and `matsnu`.

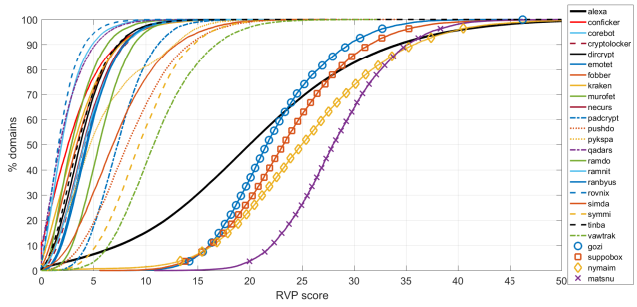


Figure 2. RVP scores for benign domains (`alexa`) and dictionary-based DGAs strongly overlap, severely impacting detection performance.

obtained by splitting domain names. Word2Vec considers both individual words and a sliding window of context words surrounding individual words as it iterates over the entire corpus of domain sentences. To generate the embedding, we apply Word2Vec using the well-known skip-gram architecture. In the continuous skip-gram architecture, the model uses the current word to predict the surrounding window of context words. The skip-gram architecture weighs nearby context words more heavily than more distant context words. The dimensionality z of the word embeddings in Word2Vec is a hyperparameter of the model. In *Dom2Vec*, we take $z = 100$, and use a sliding window of length $l = 5$ words. One of the key steps in *Dom2Vec* is the splitting of domain names into sentences. This is indeed a challenging step, as domain names often lack explicit word boundaries. In *Dom2Vec*, we approach this problem probabilistically, where we aim to find the most likely sentence that maximizes the product of the probabilities p_i of each individually identified word. The probability of a word is determined based on its frequency as observed in a learning corpora of documents D. We use in particular a publicly available dictionary of $M = 125.000$ words extracted from Wikipedia pages in English, sorted by frequency of appearance [14]. Following state of the art in NLP, we assume all words in this dictionary are independently distributed. Assuming that words follow a standard Zipf's law, the word with rank i in the dictionary has probability $p_i \approx 1/(\mathrm{i} \cdot \log M)$. In this context, the splitting of a domain name into words boils down to finding the optimal word segmentation or sentence $s_d = \{w_1, w_2, .., w_m\}$, where each word w_i represents a valid word in D. The goal is to maximize the probability $P(s_d|D) = \prod p_i$ of the sentence s_d, given the domain name d and the dictionary D.

To obtain an embedding for a domain name d, the Word2Vec embeddings for each word w_i in a sentence s_d are combined through five different aggregation approaches, including three different pooling techniques $z_d(\min)$, $z_d(\text{mean})$, and $z_d(\max)$, corresponding to the min, average, and max values for each dimension in z, the sum of the embeddings $z_d(\text{sum})$, and a weighted-sum of the embeddings $z_d(\text{TF-IDF})$, using Term Frequency-Inverse Document Frequency (TF-IDF) weighting. TF-IDF is a commonly used technique in NLP that weighs the importance of a word by considering both its frequency within a document (domain) and its rarity across the entire corpus (list of domains). By concatenating these five aggregated domain

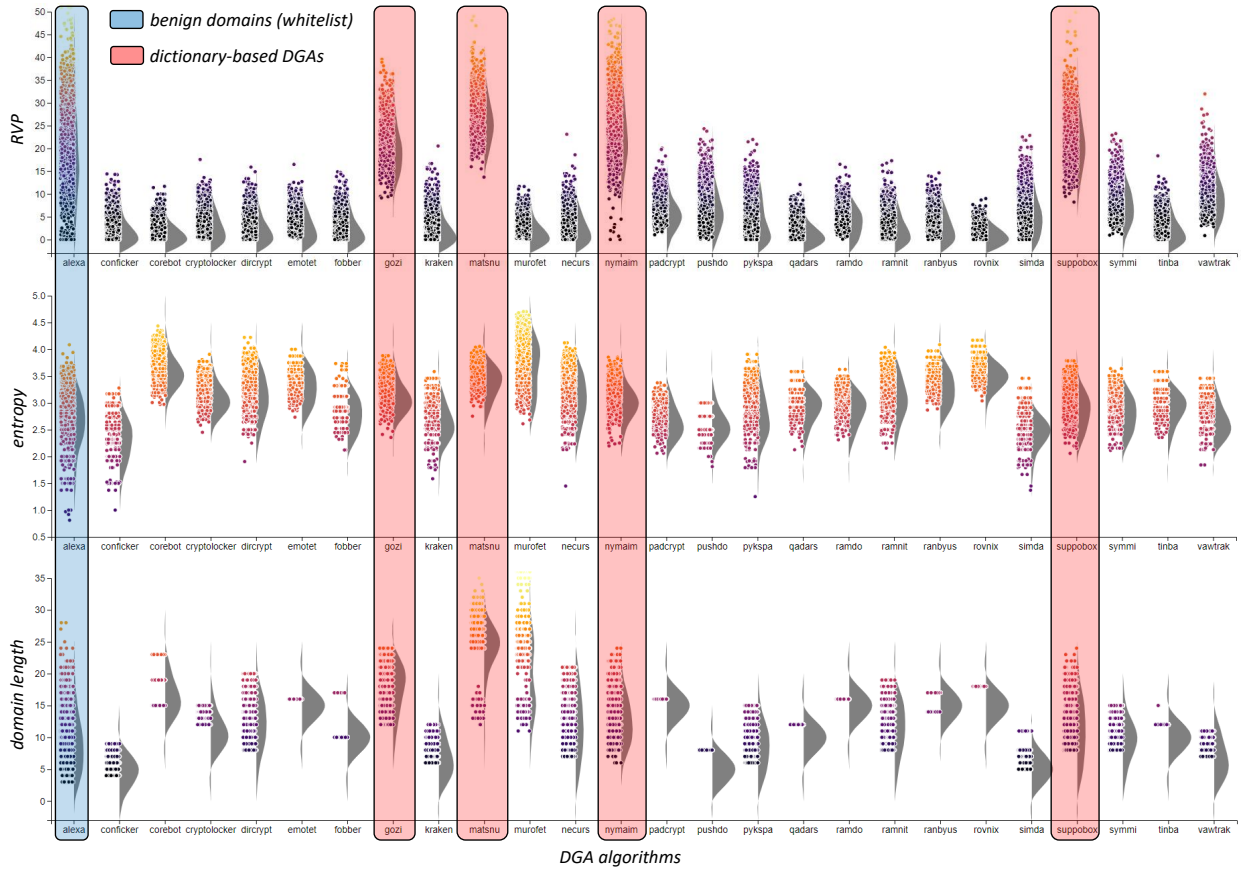


Figure 3. Empirical distribution of the input features used by RF3, for benign domains (`alexa`) and per DGA type. For ease of comparison, dictionary-based DGA algorithms are marked with boxes. DGAs such as `emotet`, `padcrypt`, or `pushdo` generate domains of constant length.

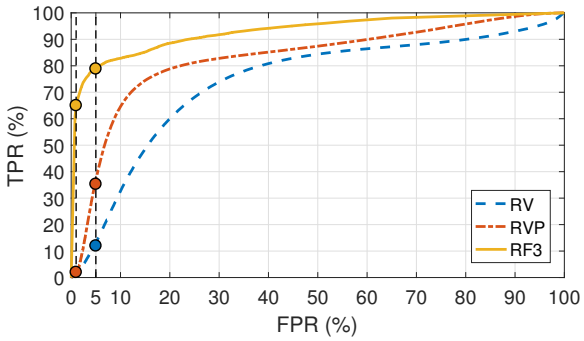


Figure 4. DGA detection performance. For the same false alarms' rate, RF3 largely outperforms RVP, particularly for FP ratios below 1%.

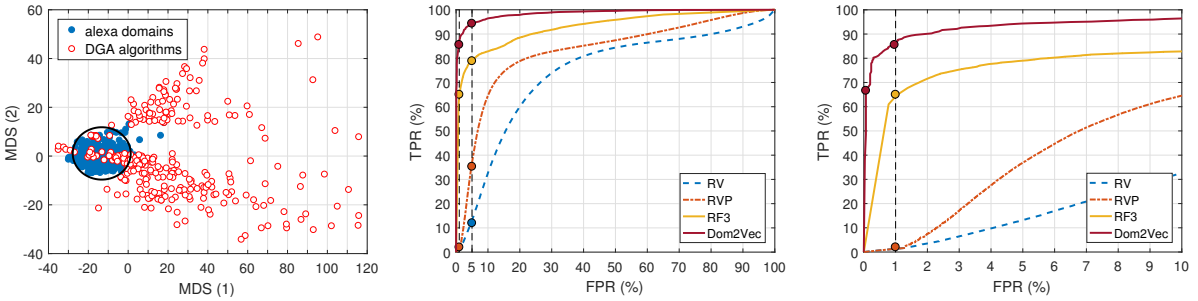
embeddings, a domain d is finally embedded into a latent space of dimension $z_d = 500$. This rich latent space is then used in supervised learning tasks.

IV. Evaluation Results

We use a publicly available DGA benchmark [1] for evaluation purposes, consisting of domains generated by 25 different DGA families from the Netlab Opendata Project repository (`https://data.netlab.360.com/dga/`), and using Alexa as an authoritative source for benign domain names. The dataset contains top-337.500 Alexa domains as whitelist, and 13.500 DGA domains per different family, resulting in a total of 675.000 domains, 50/50 balanced. We take Alexa top-500 list of domains as reputation list for RV and RVP.

Fig. 1 compares the performance of RV [8] and our RVP reputation scores, for each individual DGA algorithm i – i.e., `alexa` vs $\texttt{DGA}_i$. Firstly, RVP clearly outperforms RV for all DGA families, significantly reducing false alarms for higher detection rates. Still, RVP fails to properly detect so-called dictionary-based DGAs, referring to DGAs which use pre-defined lists of words which are similar to those used in standard domains. The list includes `gozi`, `suppobox`, `nymaim`, and `matsnu`. Fig. 2 clearly shows how the empirical distributions of RVP values for benign domains (`alexa`) and dictionary-based DGAs strongly overlap, severely impacting detection performance.

Results can be significantly improved with RF3, combining RVP(d) scores with domains length λ_d and empirical entropy $H(d)$. Fig. 3 depicts the empirical distribution of the input features used by RF3, for benign domains (`alexa`) and per DGA type. Dictionary-based DGAs are marked with boxes. We train RF3 through standard five-fold cross validation, and evaluate the normalized importance of each of the input



(a) MDS visualization of *Dom2Vec* embeddings. (b) ROC curves. (c) ROC curves, false alarm rates $\in [0, 10\%]$.

Figure 5. DGA detection with *Dom2Vec*. (a) *Dom2Vec* embeddings provide high discriminative power between normal (i.e., `alexa`) and DGA domains. Significant improvement in detection performance: (b,c) for a FPR below 1%, *Dom2Vec* can detect 32% more DGAs than RF3.

features on the classification output: while RVP clearly stands out, with a normalized impurity reduction of 72%, both λ_d and $H(d)$ capture almost 30% of the impurity decrease, evidencing how relevant they are to improve the classification power of RF3. In particular, see how both λ_d and $H(d)$ introduce higher heterogeneity to the description of a domain name d as compared to RVP – cf. Fig. 3.

Fig. 4 reports the overall DGA detection performance for RV, RVP, and RF3 in the complete dataset. RF3 largely outperforms both RV and RVP, particularly for false alarm rates below 5%. For a FPR of 5%, RF3 detects 80% of the DGA domains, falling to 35% and 12% for RVP and RV, respectively. For a more applicable FPR of 1%, RF3 detects 65% of the DGA domains, whereas RVP and RV detect only 2% of them. These results confirm that a simple approach such as RF3 can significantly boost the descriptive properties obtained with RVP, combining reputation scores with other lexicographic features for better detection performance.

Fig. 5 depicts the performance improvement introduced by *Dom2Vec* embeddings. Same as for RF3, we train a RF model for DGA detection (five-fold cross validation), using *Dom2Vec* 503 dimensional embeddings $\{z_d, \text{RVP}(d), \lambda_d, H(d)\}$ – i.e., five aggregated domain embeddings z_d, concatenated with RVP(d), λ_d, and $H(d)$. To evidence the discriminative power of *Dom2Vec*, Fig. 5(a) shows a scatter plot of the domain embeddings *Dom2Vec* – using a bi-dimensional Multidimensional Scaling (MDS) for visualization, for both benign (alexa top-337.500) and DGA-generated domains. While there is an overlapping between DGA/non-DGA domains, benign domains are significantly concentrated in a more compressed latent space than DGA domains. As a consequence, and as evidenced by the ROC curves in Fig. 5(b) and Fig. 5(c), *Dom2Vec* features realize much better detection performance. According to Fig. 5(b), for a FPR of 5%, *Dom2Vec* detects almost 95% of the DGA domains, representing an almost 20% gain as compared to RF3. More interestingly, when zooming in lower FPRs in Fig. 5(c), this gain over RF3 increases, resulting in 32% more DGAs detected for a FPR of 1% (TPR = 86%). In addition, *Dom2Vec* detects 67% of the DGAs with almost no false alarms, which is not doable with any of the other three detection approaches, including RV, RVP, and RF3.

V. Concluding Remarks

We are exploring the benefits of word embedding techniques to complement and improve DGA detection approaches, relying exclusively on the analysis of domain names. We have introduced *Dom2Vec*, a novel approach to map a domain name into an expressive, high-dimensional latent space, and shown preliminary yet promising detection results. We are currently doing a deeper study on multiple aspects of *Dom2Vec*, testing different models other than RF, assessing the impact of Word2Vec parametrization on overall performance, evaluating generalization of the approach to other datasets, testing with bigger, multi-lingual dictionaries for domain splitting, studying computational complexity, and more. **Acknowledgment**: this work has been partially supported by the Austrian FFG ICT-of-the-Future project *DynAISEC – Adaptive AI/ML for Dynamic Cybersecurity Systems* – project ID 887504.

References

[1] A. Cucchiarelli et al., "Algorithmically Generated Malicious Domain Names Detection based on N-grams Features," *Expert Sys. with Apps.*, vol. 170, 2021.
[2] H. Zhao et al., "Malicious Domain Names Detection Algorithm Based on Lexical Analysis and Feature Quantification," *IEEE Access*, vol. 7, 2019.
[3] C. M. R. d. Silva et al., "Heuristic-Based Strategy for Phishing Prediction: A Survey of URL-Based Approach," *Comput. Secur.*, vol. 88, no. C, Jan 2020.
[4] S. Maroofi et al., "Comar: Classification of compromised versus maliciously registered domains," in *2020 IEEE European S&P (EuroS&P)*, 2020.
[5] M. Korczynski et al., "Cybercrime after the Sunrise: a Statistical Analysis of DNS Abuse in new GTLDs," in *ASIACCS*, 2018.
[6] P. Mowar and M. Jain, "Fishing out the Phishing Websites," in *Conf. on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 2021.
[7] M. Korkmaz et al., "Phishing Web Page Detection Using N-gram Features Extracted From URLs," in *Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 2021.
[8] H. Zhao et al., "Malicious Domain Names Detection Algorithm Based on N-Gram," *J. Comput. Networks Commun.*, vol. 2019, 2019.
[9] T. Mikolov et al., "Efficient Estimation of Word Representations in Vector Space," in *arXiv:1301.3781 [cs.CL]*, 2013.
[10] T. Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality," in *arXiv:1310.4546 [cs.CL]*, 2013.
[11] X. Fang et al., "Domain-embeddings based dga detection with incremental training method," in *2020 IEEE Symposium on Computers and Communications (ISCC)*, 2020.
[12] W. López et al., "Learning Semantic Information from Internet Domain Names using Word Embeddings," *Engineering Applications of Artificial Intelligence*, vol. 94, 2020.
[13] L. Torrealba et al., "Phish Me If You Can - Lexicographic Analysis and Machine Learning for Phishing Websites Detection with PHISHWEB," in *9th IEEE International Conference on Network Softwarization (NetSoft)*, 2023.
[14] S. Rai et al., "Effect of Identifier Tokenization on Automatic Source Code Documentation," in *Arabian Journal for Science and Engineering*, vol. 47, 2022.