More than Words is What you Need - Detecting DGA and Phishing Domains with Dom2Vec Word Embeddings

Lucas Torrealba Aravena†, Pedro Casas*, Javier Bustos-Jiménez†, Mislav Findrik‡
lucas@niclabs.cl, pedro.casas@ait.ac.at, jbustos@niclabs.cl, mislav.findrik@cyansecurity.com
†NIC Labs, Universidad de Chile, *AIT Austrian Institute of Technology, ‡cyan Security Group

***Abstract*—The rapid detection of Domain Generation Algorithm (DGA) and general phishing domains plays a critical role in mitigating malware propagation and its potential impact, as well as in limiting botnet activity coordination through command and control (C&C) servers. We assess a learning driven approach for accurate detection of DGA-generated and phishing domains, leveraging word embeddings learned from observed domain names in DNS queries or browsing URLs. Domain embeddings are constructed with *Dom2Vec* (D2V), a novel technique which builds on top of word embedding models (e.g., *Word2Vec*) to map words and tokens extracted from domain names into highly expressive representations. Through experimental evaluation on a large-scale dataset of almost 800,000 domains, comprising 25 distinct families of DGA domains and general phishing URLs, we demonstrate the goodness of D2V embeddings for phishing detection, in particular for the detection of DGAs.**

***Index Terms*—Phishing Detection, Word2Vec, TF-IDF, *Dom2Vec* (D2V).**

I. Introduction

Domain Generation Algorithms (DGAs) have become prevalent in malware to establish and maintain a Command and Control (C&C) infrastructure [7]. Botnets heavily rely on C&C servers to coordinate bots, i.e., compromised machines. To evade detection, botnets often employ DGAs that generate a diverse set of (quasi) random domain names based on a seed parameter, sometimes relying on pre-defined dictionaries [3]. By employing a shared algorithm, botmasters can register the C&C server on the network for a short duration with a randomly selected DGA domain name, allowing it to hide behind different domain names at different times. 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 [5], etc. Blocklisting only requires having access to the domain name, eliminating the need for external information sources. This is highly convenient, especially in terms of speed of analysis and large-scale application, but also in terms of privacy preservation for end-users, as the content of the domain itself is never accessed. While blocklisting is efficient and simple to implement and interpret, there are clear limitations on its application in the practice, including the lack of protection against newly created sites (i.e., zero-day phishing attacks), the costs and effort to keep correctly updated lists, as well as the accuracy with which information is registered within a blocklist – e.g., if a single character is changed, it becomes a totally different domain. Therefore, blocklisting is generally adopted as a first defense line to protect users from well-known phishing attacks.

A large literature on detection of malicious websites and DGAs [6] has been devoted to the conception of analysis heuristics and machine learning-driven approaches to extract features, keywords, and patterns from the lexicographic analysis of domain names, which better correlate to the occurrence of malicious activity. Most approaches in the literature are based on the computation of features, derived from n-gram tokens of the domain name [1], [3], [4], [8], [9]. While commonly extracted features can differentiate between DGA and non-DGA domains [4], [9], [12], we have shown that they tend to be less effective for highly-accurate detection, failing in some cases to detect DGAs 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 propose a novel approach to DGA and phishing detection from the analysis of the domain name itself, exploiting the power of word embeddings and machine learning models. We introduce a learning based approach leveraging Word2Vec [10], [11] to embed domain names into a highly expressive latent space, which allows for better detection of DGAs and phishing domains. 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. We use in particular *Dom2Vec* (D2V), an approach we have conceived to map domain names into high dimensional vectors, and use them to train simple ensemble learning models for binary detection of DGAs and general phisihing domains.

We assess the performance of this proposal in the detection of DGA and more general phishing domains, using two publicly available datasets: (1) a *DGA-dataset*, consisting of well-known domains considered as benign (we take top Alexa domains) and a list of DGAs, generated out of 25 different DGA families [3]; (2) a modified version of *PHISHSTORM*

978-3-903176-64-5

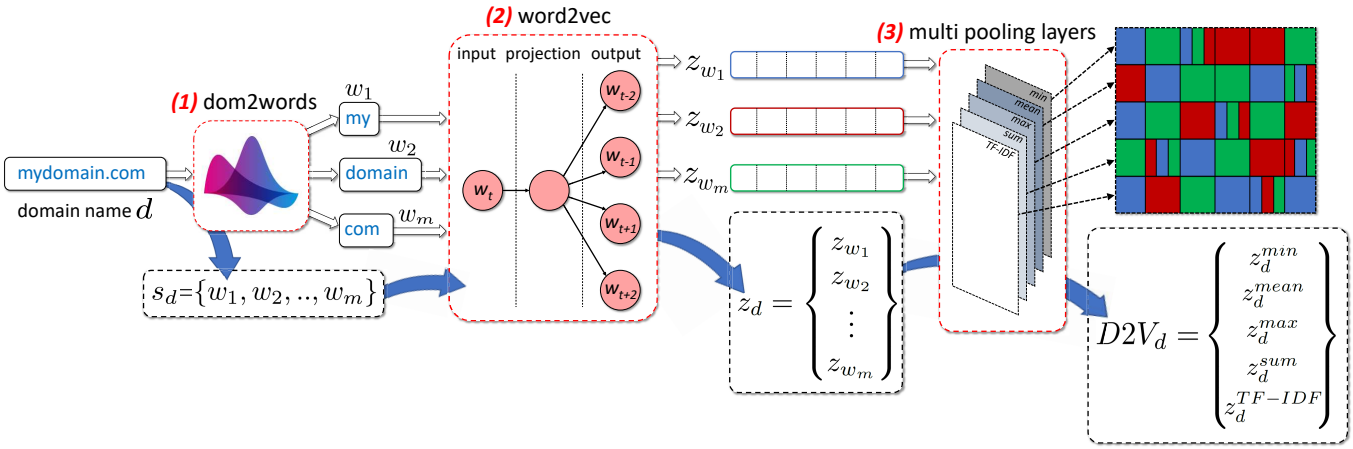


Figure 1. *Dom2Vec* embedding pipeline. *Dom2Vec* embeds a domain name d into a dense, highly expressive latent space D2V_d. The pipeline includes a (i) domain-to-words step, where d is split into a set of *known words* or tokens w_i, (ii) word2vec embedding of each word w_i into z_{w_i}, and (iii) five independent pooling layers to merge and down sample the set of embeddings z_{w_i} into a fix-length representation D2V_d.

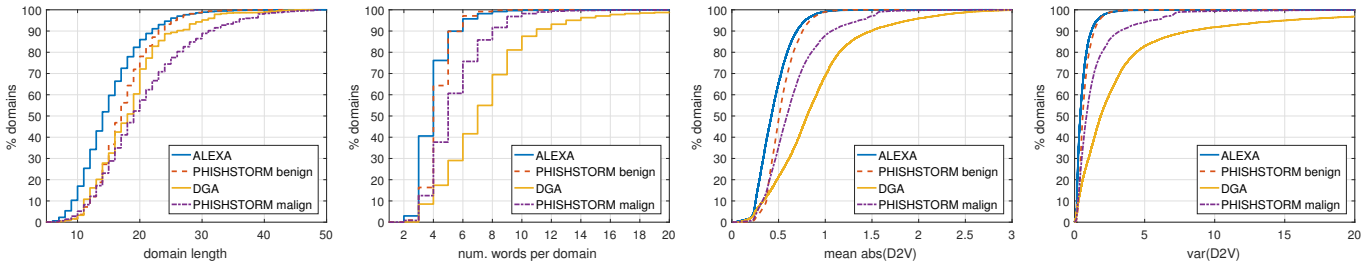
[2], a well-known list of almost 100,000 phishing and legitimate URLs obtained from both PhishTank (malign) and open lists of browsing URLs (benign).

II. Dom2Vec - Word2Vec for Domain Analysis

Dom2Vec leverages the power of Word2Vec to generate embeddings for domain names, where DGAs and benign domain names are significantly different. Fig. 1 presents a diagram of the embedding pipeline followed in *Dom2Vec*. 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 into 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'}. We refer to this domain splitting step as *dom2words*. Word2Vec is then applied on top of the resulting sentences $s_{d_i} = \{w_{j,i}\}$, obtaining as such an embedding $z_{w_{j,i}}$ for each word $w_{j,i}$ observed in a training dataset. Finally, an embedding for domain d is computed out of the embeddings z_{w_j} of each of the words w_j 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.

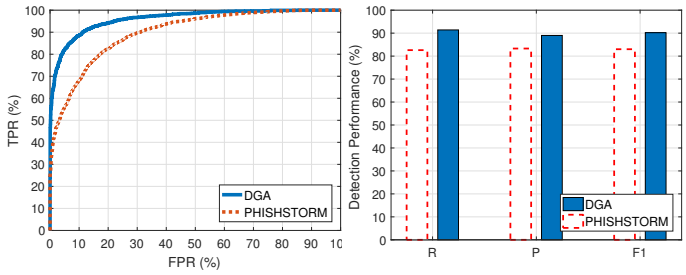
The *dom2words* splitting of domain names into sentences is a challenging step, as domain names often lack explicit word boundaries. In *Dom2Vec*, we approach this problem probabilistically, where we aim to find the most likely sentence that maximizes the product of the probabilities p_i of each individually identified word. The probability of a word is determined based on its frequency as observed in a learning corpus 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 $p_i \approx 1/(\text{i} \cdot \log M)$. 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.

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 an 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 γ of the word embeddings in Word2Vec is a hyperparameter of the model. In *Dom2Vec*, we take $\gamma = 100$, and use a sliding window of length $l = 5$ words. To obtain a final embedding D2V_d for a domain name d, the Word2Vec embeddings z_{w_i} for each word w_i in a sentence s_d are combined through five different aggregation



(a) Domain length (b) Words per domain (c) mean(D2V) (d) var(D2V)

Figure 2. Characterizing benign domains (Alexa top-337.500), PHISHSTORM benign, DGA, and PHISHSTORM phishing.



(a) ROC detection performance (b) Detection performance

Figure 3. A D2V-based detector detects about 85% of the DGAs and 60% of the PHISHTORM phishing URLs with a FPR below 5%. F1-scores of 0.9 and 0.82 are realized in the complete assessment of both datasets.

approaches, including three different pooling techniques z_d^{min}, z_d^{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^{sum}, and a weighted-sum of the embeddings z_d^{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 embeddings, a domain d is finally embedded into $\text{D2V}_d = \{z_d^{min}, z_d^{mean}, z_d^{max}, z_d^{sum}, z_d^{TF-IDF}\}$, with a latent space dimension $\text{D2V}_d \in \mathcal{R}^{500}$. This rich latent space is then used in a supervised learning task for domain classification.

III. Evaluation Results

To study *Dom2Vec* embeddings and other features, and to evaluate DeepD2V, we take a publicly available DGA benchmark [3]. This benchmark consists of domains generated by 25 different DGA families from the Netlab Opendata Project repository (https://data.netlab.360.com/dga/), using Alexa as an authoritative source for benign domain names. The list of DGAs includes well known malware and crypto-related DGAs (conficker, kraken, cryptolocker, etc.), as well as specific dictionary based DGAs (gozi, matsnu, nymaim, and suppobox). The dataset contains top-337.500 Alexa domains as benign domains, and 13.500 DGA domains per different family, resulting in a total of 675.000 domains, 50/50 balanced. We also use PHISHSTORM [2], a list of almost 100,000 phishing and legitimate URLs for analysis. Malign URLs are based on commonly used techniques in phishing crafting, including typosquatting and obfuscation. As D2V operates on top of domain names, we keep only the domain names associated to these malign URLs. As we see next, while this certainly degrades the quality of the dataset (and therefore the performance of D2V) – i.e., a URL can be phishing, but not its associated domain name, it allows for a combined analysis of both datasets.

A. Detection Performance and Interpretation of Results

We address the DGA and phishing detection as a binary classification task, and train a LightGBM-based model using a 80/20 random split for training plus validation, and testing purposes. To evaluate model generalization and overfitting of the training, we follow standard five-fold cross validation in the training set, using early stopping to identify the target performance in the final training step – i.e., training on the complete training set. Results presented next corresponds to the detection performance in the testing set.

The combination of D2V embeddings with a LightGBM model provides good detection performance, achieving high recall and precision for both DGAs and PHISHSTORM, with $F1$ scores of 0.9 and 0.82, respectively. Fig. 3(a) reports the realized detection performance, in terms of ROC curves, and Fig. 3(b) summarizes detection performance results for both datasets. Note that the high performance observed for DGA domains is likely due to the inherent characteristics of these algorithmically generated names. Their repetitive patterns and use of specific character sets make them easier for D2V embeddings to identify. Detection of general phishing domains, however, presents a bigger challenge. While D2V embeddings capture semantic relationships within the domain name, phishers often employ techniques to obfuscate their malicious intent. This can include misspelling legitimate brand names, using subdomains, or incorporating special characters. In addition, only keeping domain names from the PHISHSTORM URLs list makes the labeling of the truncated somehow inconsistent, which is probably also impacting the lower detection performance in this datatset. In our future work, we plan to explore incorporating additional information to the D2V embeddings' space, to further improve detection accuracy for these more general phishing attempts.

ACKNOWLEDGMENTS

This work has been supported by the Austrian FFG ICT-of-the-Future project *DynAISEC* – Adaptive AI/ML for Dynamic Cybersecurity Systems – project ID 887504.

REFERENCES

[1] S. Yadav, A. Reddy, N. Redd, and S. Ranjan, "Detecting Algorithmically Generated Malicious Domain Names," in *ACM Internet Measurement Conference (IMC)*, 2021.

[2] S. Marchal, J. Francois, R. State, and T. Engel, "Phishstorm: Detecting phishing with streaming analytics," *IEEE Transactions on Network and Service Management*, vol. 11, no. 4, pp. 458–471, 2014.

[3] 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, 2021.

[4] J .Selvi, R .Rodríguez, E .Soria-Olivas, "Detection of Algorithmically Generated Malicious Domain Names using Masked N-grams," *Expert Systems with Applications*, vol. 124, 2019.

[5] H. Zhao et al., "Malicious Domain Names Detection Algorithm Based on Lexical Analysis and Feature Quantification," *IEEE Access*, 2019.

[6] C. Revoredo da Silva et al., "Heuristic-Based Strategy for Phishing Prediction: A Survey of URL-Based Approach," *Comput. Secur.*, 2020.

[7] S. Maroofi, M. Korczynski, C. Hesselman, B. Ampeau, and A. Duda, "COMAR: Classification of Compromised versus Maliciously Registered Domains," in *IEEE EuroS&P*, 2020.

[8] 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.

[9] H. Zhao et al., "Malicious Domain Names Detection Algorithm Based on N-Gram," *J. Comput. Networks Commun.*, 2019.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *arXiv:1301.3781*, 2013.

[11] T. Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality," in *arXiv:1310.4546*, 2013.

[12] L. Torrealba, P. Casas, J. Bustos-Jiménez, G. Capdehourat, and M. Findrik, "Phish Me If You Can - Lexicographic Analysis and Machine Learning for Phishing Websites Detection with PHISHWEB," in *9th IEEE Int. Conf. on Network Softwarization (NetSoft)*, 2023.

[13] L. Torrealba, P. Casas, J. Bustos-Jiménez, G. Capdehourat, and M. Findrik, "Dom2Vec - Detecting DGA Domains through Word Embeddings and AI/ML-driven Lexicographic Analysis," in *19th IEEE International Conference on Network and Service Management (CNSM)*, 2023.

[14] S. Rai, R. Belwal, and A. Gupta, "Effect of Identifier Tokenization on Automatic Source Code Documentation," in *Arabian Journal for Science and Engineering*, vol. 47, 2022.