Optimizing Phishing URL Detection: A Comparative Analysis of Feature Counts and Training Times

Moayyad Alajlouni, Yahya Tashtoush, Saud Alhajaj ALDossari, and Omar Darwish
Department of Computer Science, *Jordan University of Science and Technology*, Irbid, Jordan
moaead-alajlouni@hotmail.com
Department of Computer Science, *Jordan University of Science and Technology*, Irbid, Jordan
yahya-t@just.edu.jo
Department of Electrical Engineering, Prince Sattam bin Abdulaziz University, Wadi Addwasir, Saudi Arabia
s.alhajaj@psau.edu.sa
Information Security and Applied Computing Department, *Eastern Michigan University, Ypsilanti*, MI 48197, USA
odarwish@emich.edu

***Abstract*—Phishing attacks are the biggest cybersecurity threats in the digital world. Attackers exploit users by impersonating real, authentic websites to obtain sensitive information such as passwords and bank statements. One common technique used in these attacks is malicious URLs. These malicious URLs mimic legitimate URLs, misleading users into interacting with malicious websites. This practice, URL phishing, presents a big threat to internet security, emphasizing the need for advanced detection methods.**

This paper presents a method for detecting malicious URLs using statistical features extracted from n-grams. These n-grams are extracted from the hexadecimal representation of URLs. We used Explainable AI (XAI) to explore the extracted features and evaluate their importance and role in phishing detection. A key advantage of our method is its ability to reduce the required features and the training time by using fewer features after applying XAI techniques. Our technique only uses statistical features extracted from n-grams and the n-gram itself, without requiring any high-level features.

We applied XAI techniques, SHapley Additive exPlanations (SHAP), and Local Interpretable Model-agnostic Explanations (LIME). Through the explanation provided by XAI methods, we were able to determine the most important features in our feature set, enabling a reduction in feature count. Using fewer features (4, 7, 10, 13, 15), we got good accuracy compared to the 41 features and 87 features used in the original experiment and reduced the models' training times and complexity.

This research aimed to enhance phishing URL detection by optimizing feature selection and analyzing the impact on training times. Our findings show the importance of using minimal statistical features to identify malicious URLs effectively and efficiently.

***Index Terms*—N-gram, Feature selection, URL-based Features, Low-Level Features, High-Level Features, Convolutional Neural Network (CNN), IQR, Explainable AI (XAI), SHAP, LIME, URL**

I. Introduction

A. Overview

Phishing through URLs, commonly referred to as "website phishing," represents a type of internet scam where cybercriminals create counterfeit websites or harmful links that closely resemble legitimate ones. The aim is to trick users into divulging sensitive details, like their login information or financial data [1]. Deceptive emails, social media posts, or online advertisements are frequently used to lure victims of URL phishing into accessing these bogus websites.

Phishing occurs through a variety of channels, including the Internet, Short Messaging Service (SMS), and phone. Email, instant messaging, voice phishing, and websites can all be targets [2]. Although the carefulness and experience of the user are important, it is not possible to completely prevent users from falling victim to phishing scams [3].

According to the Anti-Phishing Working Group (APWG) 3rd Quarter, 2018 report [4], the total number of detected phishing attacks was 151,014. One-third of all phishing efforts in 2013 attempted to access bank accounts or other financial data, according to research. From 2011 to 2012, financial phishing attempts increased by 8.5%, reaching an all-time high [5]. Currently, phishing ranks among the most profitable forms of cybercriminal endeavor. In a 2007 report, Gartner Research estimated that phishing schemes caused a financial impact of $3.2 billion [6]. Machine learning techniques have been used to investigate the URLs of web pages with different feature sets in order to refine detection efficiency [7] [8].

The contributions outlined in this paper are:

- A detection method was developed utilizing only n-gram features and representations.
- Multiple machine learning models were employed, showcasing the versatility of our methods across various algorithmic approaches.
- Explainable Artificial Intelligence (XAI) techniques, namely SHAP and LIME, were utilized to assess and validate the significance of selected features, thereby enhancing model transparency and facilitating better comprehension.

979-8-3503-5469-0/24/$31.00 ©2024 IEEE

- Optimization was performed to strike a balance between feature count, model complexity, and detection accuracy, a critical aspect for real-time applications.
- The method effectively reduced dependence on a high number of high-level URL features, streamlining the detection process.

B. Uniform Resource Locator Structure (URL)

The Uniform Resource Locator (URL), as defined in RFC1738 [9], is a string representation of an internet resource. The following is an example of a URL:

<scheme>:<scheme-specific-part>

The scheme indicates the resource's access method or network location (for example, http, ftp, mailto), whereas the remainder of the URL may differ based on the scheme used. The following syntactic formulation for the HTTP protocol may be:

<host>:<port>/<URL-path>

It begins with a network host's domain name or IP address. Then there's the port number to connect to and the URL path that describes how to access the resource (for example, *http://host.com:80/page*).

C. Phishing Detection

Phishing URL detection can be accomplished using either anticipatory or responsive methods. One example is the Google Safe Browsing API 3. Such programs generate a list of potentially hazardous URLs for investigation. These lists are created through a variety of techniques, including manual submission, the use of honeypots, and the search for known phishing characteristics on the internet [10], [11]. Web browsers, for example, use these lists to restrict access to URLs found on them. One downside of this responsive approach is that a phishing URL must first be added to the blacklist before it can be banned. Users are exposed until the URL is reported and the list is updated. Furthermore, because most phishing websites have a lifespan of less than 24 hours [11], [12], by the time they are placed on the blacklist, their goal has been accomplished.

Proactive strategies address this issue by instantly examining the features of a webpage to assess its potential danger. A classification model [13] assists in this evaluation. Techniques such as support vector machines [14], real-time analytics [15], gradient boosting [16], [17], random forests [18], latent Dirichlet allocation [19], continuous incremental learning [20], and neural networks [21] are employed for phishing detection. Many of these strategies rely on a collection of webpage properties, suggesting that a site must be fully loaded before the algorithm can be applied. This causes a significant delay in the evaluation [22], [23].

D. Low-Level Features and N-Grams

N-grams, or contiguous sequences of n elements, are derived from text or voice data. The "N-gram" prefix, which indicates the number of components in a sequence, includes unigrams (1 gram), bigrams (2 gram), trigrams (3 gram), and so on. Figure 1 illustrates the process of creating 4-grams from a given text. Low-level features, like N-grams generated from hexadecimal representations of URLs, capture intricate patterns and variations in character sequences. They provide insights into the underlying structure of URLs. On the other hand, high-level aspects, such as URL length, sub-domain count, and special character usage, offer a broader perspective on URL structure and semantics.

By analyzing the patterns employed by malicious websites, N-grams aid in identifying and detecting phishing attempts [24]. In the realm of phishing detection, these N-grams provide valuable insights into patterns that malicious websites may employ, thereby aiding in their identification.

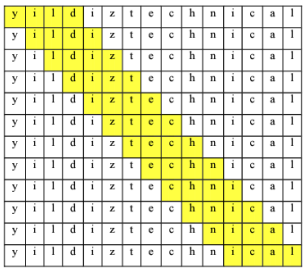

Fig. 1. Example of 4-gram Segmentation

II. High-Level vs. Low-Level Features

A. High-Level Features

High-level features focus on the general standards of a webpage while offering broad insights about URLs. They might include information on the domain's registration or the presence of SSL certificates. These characteristics encompass its overall attributes.

B. Low-Level Features

On the other hand, low-level features delve into further complexity and are frequently derived from separate URL sections or components. Instead of focusing on the URL's general properties, these qualities are obtained by examining its specific components. The n-grams of the URL are crucial for extracting low-level features in this study. The statistical data about the URL's structure is captured by each of the features, revealing patterns often exploited by malicious entities. These features include mean, median, standard deviation, variance, minimum, 25th percentile, 75th percentile, IQR, maximum, skewness, kurtosis, entropy, and other statistical metrics and features described in later sections.

The primary action in our methodology involved transforming the URL into its corresponding hexadecimal representation. By doing this, we could depict the URL consistently and mathematically, paving the way for subsequent feature extraction. After we created a hexadecimal representation and extracted n-grams from it, we extracted statistical features from these n-grams.

III. N-GRAMS

In text analysis, an n-gram refers to a sequential grouping of n elements derived from a given text or sentence sample. The value of n determines the length of the sequence. For instance, when n equals 2, it is specifically known as a bigram [25]. In our analysis, we convert URLs to their hexadecimal representation before extracting n-grams from these hexadecimal strings.

After extracting hexadecimal strings, we extract n-grams from these strings. For example, when extracting 2-grams (bigrams), we generate sequences such as ”68”, ”87”, and ”74”, sequentially moving through the string. For 4-grams, the extracted sequences look like ”6874”, ”8747”, ”7474”, etc. We proceed through the string in sequence. This pattern of extraction can be extended for longer sequences such as 6-grams, 8-grams, and more lengths.

IV. MACHINE LEARNING MODELS

Since the dawn of time, humans have utilized a variety of tools to make tasks simpler to perform. Machine Learning is the scientific study of algorithms and statistical models that computer systems employ to perform a specific task without explicit instructions. It is a technique for teaching machines to better handle data [26]. Machine learning is one of the subsections of artificial intelligence that deals with algorithms [27]. Machine learning refers to the process of developing computer algorithms that can mimic human intelligence. It incorporates concepts from artificial intelligence, probability and statistics, computer science, information theory, psychology, control theory, philosophy, and other areas. These algorithms are utilized in a wide range of applications, such as data mining, image processing, and predictive analytics [28] [29].

The main advantage of using machine learning is that once an algorithm learns how to handle data, it can complete tasks independently. A smart algorithm, like humans, can learn ”lifelong” as it analyzes fresh data and learns from its failures [30].

A. Explainable Artificial Intelligence (XAI)

As machine learning and deep learning models become more complex, the need for understanding their decision-making processes has given rise to the field of XAI. XAI aims to bridge the gap between human understanding and AI robustness by explaining how these models work. The main idea of XAI is to provide explanations that humans can understand. These explanations can highlight the importance of features in visualizing the internal decision-making of models. LIME and SHAP have been developed to achieve explainability in AI models [31].

1) LIME: The key advantage of LIME is its ability to offer explanations for individual predictions, allowing users to understand model behavior on a case-by-case basis [32].

2) SHAP: SHAP provides a global perspective on feature importance, offering a view of how each feature impacts the model’s decisions across the entire dataset [32].

V. RELATED WORK

In this study, we used the dataset created by Hannousse et al., and we compared the results using just URL-based features. They explored the creation of benchmark datasets, which are essential for assessing machine learning models when it comes to identifying phishing websites. After carefully classifying and analyzing 87 distinct features, they focused on three primary categories: external, content, and URL-based. One of their study’s main conclusions, which we also saw in our work, was the effectiveness of URL-based features. They achieved 91.03% accuracy using URL-based features; however, the external-based features came in first place with a slightly higher percentage of 94.09%. Content-based features had the lowest accuracy at 89.87% [33].

Sameen et al. introduced a model that achieved 98% accuracy through the analysis of 17 distinct features extracted from URLs. This model stands out for its ability to efficiently parse and assess various components of a URL. Unlike traditional methods that heavily rely on third-party services or comprehensive source code analysis, Sameen et al.’s approach focuses on the intrinsic features of URLs [34].

Twelve features were employed by Marchal et al. based on URL popularity and intra-URL relatedness. Firstly, natural language processing was used to analyze web page URLs to extract important embedded terms. Using similarity metrics, intra-URL relatedness is retrieved from the extracted word set. Additionally, the terms are used to query Google and Yahoo search engines to find aspects related to URL popularity [35].

The authors delve into the efficacy of CNN for embedding characters in cybersecurity applications, specifically focusing on the identification of malicious URLs, file paths, and registry keys. Their innovative approach, which leverages character-level embeddings alongside CNN architectures, showcased exceptional performance in discerning malicious URLs, as evidenced by an impressive Area Under the Curve (AUC) value of 0.993 [36].

Term Frequency and Inverse Document Frequency (TF-IDF) based features retrieved from URLs were employed by Rao et al. in conjunction with a Random Forest classifier trained on 35 URL-based features. The accuracy score obtained by the model was 98.25% [37].

Jain and Gupta employed three third-party features and fourteen URL-based features. Within two distinct training data sizes, they tested with two machine learning classifiers. Support Vector Machine (SVM) received the best accuracy score of 91.28% [38].

VI. MATERIALS & METHODOLOGY

A. Dataset

In this study, we use the dataset from the research by Hannousse, Abdelhakim, and Salima Yahiouche in their paper ”Towards benchmark datasets for machine learning based website phishing detection: An experimental study” [33]. The dataset was constructed as a benchmark for evaluating machine learning-based website phishing detection systems. The dataset

has 87 features, categorized into various classes. These features include features gathered from external services, features extracted directly from web page contents, and it has also URL-based features.

In our work, we focus only on using the URL column and the status column from this dataset. The URL column is used to convert the URLs into hexadecimal representation, which is then employed to create different n-gram lengths. These n-grams are utilized to extract URL-based statistical features and also to use these n-grams directly with deep learning models, forming the basis of our phishing detection analysis. The tables below show a summary of the dataset.

TABLE I
SUMMARY OF THE PHISHING DATASET

Characteristic	Description
Number of Instances	11430
Number of Features	89
Feature Types	URL Characteristics, Hostname Length, IP Address, Number of Dots, Hyphens, Special Characters, Status (Legitimate/Phishing)
Sample Features	url, length_url, length_hostname
Target Variable	status

TABLE II
CLASS DISTRIBUTION IN THE PHISHING DATASET

Class	Frequency
Legitimate	5715
Phishing	5715

VII. RESULTS & DISCUSSIONS

A. Evaluation Metrics

We used the Accuracy metric to compare and measure the success of each model. Since our dataset is balanced, accuracy is particularly effective for evaluating model performance, it gives a true reflection of the model's ability to correctly predict outcomes across all classes equally. The accuracy of a model is determined by its ability to find correlations and patterns between variables as shown in the formula (1) [39]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

Where: TP (True Positives), TN (True Negatives), FP (False Positives), FN (False Negatives).

B. Feature Extraction

In this study, we extracted a total of 41 features from the URLs, derived from various statistical, positional, and character-based features of n-grams. The table below lists all the features used in our analysis.

TABLE III
EXTRACTED FEATURES FROM URLS

Feature	Description
mean	Mean value of the n-gram integers
median	Median value of the n-gram integers
std	Standard deviation of the n-gram integers
variance	Variance of the n-gram integers
min	Minimum value of the n-gram integers
25%	25th percentile of the n-gram integers
75%	75th percentile of the n-gram integers
iqr	Interquartile range of the n-gram integers
max	Maximum value of the n-gram integers
skew	Skewness of the n-gram distribution
kurtosis	Kurtosis of the n-gram distribution
entropy	Entropy of the n-gram distribution
freq	Frequency of the most common n-gram
uniq_ratio	Ratio of unique n-grams to total n-grams
entropy_rate	Normalized entropy per length of n-gram sequence
rare_ngrams	Proportion of n-grams that occur only once
common_ngrams	Proportion of n-grams that occur more than once
position_mean	Average position of n-grams in a URL
position_std	Standard deviation of n-gram positions in a URL
combination_ratio	Diversity of consecutive n-gram combinations
avg_freq	Average frequency of n-grams
max_freq	Maximum frequency of n-grams
ngram_variability	Variance in the frequency of different n-grams
first_ngram	The first n-gram in the URL sequence
last_ngram	The last n-gram in the URL sequence
char_count_0	Count of character '0' in n-grams
char_count_1	Count of character '1' in n-grams
char_count_2	Count of character '2' in n-grams
char_count_3	Count of character '3' in n-grams
char_count_4	Count of character '4' in n-grams
char_count_5	Count of character '5' in n-grams
char_count_6	Count of character '6' in n-grams
char_count_7	Count of character '7' in n-grams
char_count_8	Count of character '8' in n-grams
char_count_9	Count of character '9' in n-grams
char_count_a	Count of character 'a' in n-grams
char_count_b	Count of character 'b' in n-grams
char_count_c	Count of character 'c' in n-grams
char_count_d	Count of character 'd' in n-grams
char_count_e	Count of character 'e' in n-grams
char_count_f	Count of character 'f' in n-grams

C. Result Analysis

This subsection outlines the results of the accuracies obtained using 41 features extracted from n-gram representations:

TABLE IV
ACCURACY OF MACHINE LEARNING MODELS WITH 41 FEATURES

Model	2-grams	4-grams	6-grams	8-grams
Logistic Regression	75.73%	67.77%	50.02%	50.02%
Random Forest	83.78%	83.34%	84.13%	84.43%
AdaBoost	81.24%	82.82%	83.38%	83.47%
XGBoost	84.56%	86.10%	87.14%	**88.15%**
Naive Bayes	62.75%	66.55%	54.35%	55.62%
Neural Network	71.27%	49.98%	49.98%	49.98%
Decision Tree	74.68%	80.50%	79.93%	80.67%
LightGBM	85.05%	86.62%	87.28%	87.84%
Gradient Boosting	79.84%	82.12%	82.38%	83.52%
CatBoost	80.85%	82.82%	82.99%	82.03%
ExtraTrees	82.68%	82.77%	82.68%	82.47%

D. Explainable AI Analysis with SHAP and LIME

Our research not only evaluates machine and deep learning models for phishing detection but also delves into the interpretability of these models. We used XAI methods—specifically, SHAP and LIME. SHAP gives us a global perspective of how each feature influences the model's predictions, while LIME provides local explanations for specific instances.

1) Global Interpretability with SHAP: We used SHAP to understand the feature importance in the model. The SHAP summary plot (Figure 2) shows that the 'max' feature, representing the highest n-gram frequency within a URL, is the most powerful factor in the prediction process and is followed by the counts of specific hexadecimal characters such as 'char_hex_3_count', 'char_hex_d_count', and 'char_hex_0_count'.

2) Local Interpretability with LIME: Based on the insights provided by SHAP, we analyzed individual predictions using LIME to reveal the localized reasoning behind the model's classifications. For instance, as illustrated in Figure 3, LIME explicates the prediction of an instance as non-phishing, with features like 'max' and 'char_hex_a_count' significantly influencing the model decision.

Conversely, Figure 4 shows the LIME explanation for an instance predicted as phishing and also shows that the 'max' feature is important in the classification. This instance-specific analysis complements the broader SHAP results, enabling us to correlate and validate the global feature importance with local decision-making patterns.

SHAP and LIME analyses (Figures 2, 3, and 4) give us more understanding of the model's behavior. Together, they confirm that features like 'max' and specific hexadecimal character counts are consistently significant in distinguishing phishing URLs.

3) Feature Selection and Speed-Up Analysis: Through a series of experiments, we investigated the impact of feature set size on training time for different classifiers to identify an optimal balance between model complexity and computational

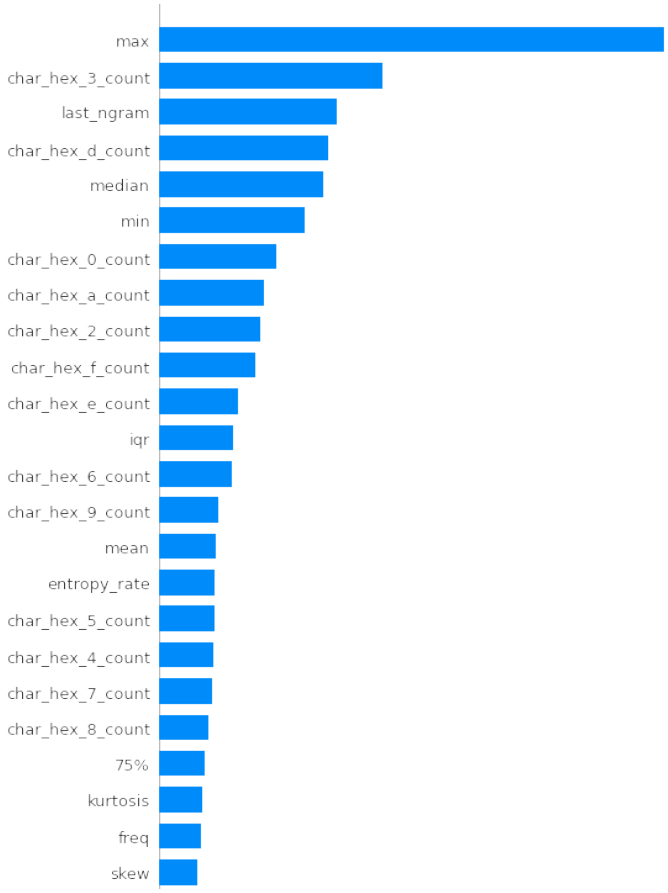

Fig. 2. SHAP Summary Plot for XGBoost Model

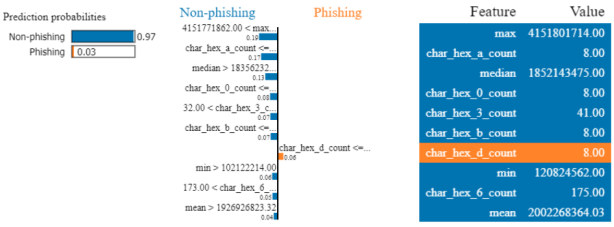

Fig. 3. LIME Explanation for Non-phishing Prediction

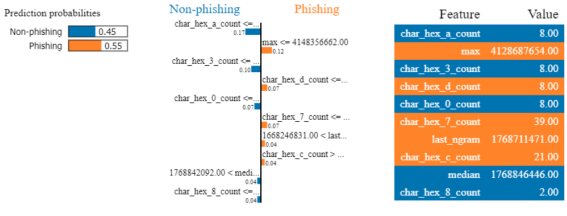

Fig. 4. LIME Explanation for Phishing Prediction

efficiency. We compared the training times using our 41 extracted features against various smaller feature sets. The feature sets of sizes 4, 7, 10, 13, and 15 were chosen based on their SHAP importance scores from our previous experiments.

We observed that increasing the number of features generally resulted in longer training times, although not linearly. Some classifiers, such as Logistic Regression and Decision Tree, showed significant speed improvements with fewer features, while others, like Random Forest and AdaBoost, exhibited moderate improvements.

4) Feature Selection and Speed-Up Analysis: Through a series of experiments, we investigated the impact of feature set size on training time for different classifiers to identify an optimal balance between model complexity and computational efficiency. We compared the training times using our 41 extracted features against various smaller feature sets. The feature sets of sizes 4, 7, 10, 13, and 15 were chosen based on their SHAP importance scores from our previous experiments.

We observed that increasing the number of features generally resulted in longer training times, although not linearly. Some classifiers, such as Logistic Regression and Decision Tree, showed significant speed improvements with fewer features, while others, like Random Forest and AdaBoost, exhibited moderate improvements.

The following table presents a detailed comparison of the speed-up in training times for various feature sets across different models, relative to the original 87 features from the paper we compare with. This comparison underscores the computational benefits of using smaller feature sets without substantially compromising accuracy.

TABLE V
COMPARISON OF SPEED-UP IN TRAINING TIMES FOR VARIOUS FEATURE SETS ACROSS DIFFERENT MODELS

Classifier	4 Features	7 Features	10 Features	13 Features	15 Features	41 Features
Logistic Regression	31x faster	31x faster	15.5x faster	15.5x faster	15.5x faster	15.5x faster
Random Forest	2.95x faster	2.78x faster	2.20x faster	2.02x faster	1.88x faster	1.03x faster
AdaBoost	6.02x faster	4.49x faster	4.02x faster	3.12x faster	2.57x faster	1.09x faster
XGBoost	4.69x faster	4.28x faster	3.79x faster	3.23x faster	2.63x faster	1.25x faster
Naive Bayes	3x faster	3x faster	3x faster	3x faster	3x faster	3x faster
Decision Tree	25x faster	15x faster	12.5x faster	8.33x faster	6.82x faster	2.78x faster
LightGBM	2.23x faster	2.03x faster	2.03x faster	1.97x faster	1.57x faster	1.23x faster
Gradient Boosting	9.75x faster	6.75x faster	5.85x faster	4.33x faster	3.38x faster	1.31x faster
CatBoost	2.79x faster	2.70x faster	2.70x faster	2.89x faster	2.89x faster	2.19x faster
ExtraTrees	2.04x faster	1.64x faster	1.19x faster	1.15x faster	1.12x faster	1.55x faster

VIII. CONCLUSIONS

This research aimed to optimize phishing URL detection through a comparative analysis of feature counts and training times. Our findings demonstrate the computational benefits and efficiency of using a reduced number of features without substantially compromising accuracy. By analyzing the impact of various feature sets, we identified an optimal balance between model complexity and computational efficiency. The results show that using smaller feature sets can significantly reduce training times while maintaining high detection accuracy, underscoring the importance of feature selection in enhancing phishing URL detection systems.

IX. FUTURE WORK

Future work will focus on expanding the analysis with new features and datasets to further validate the robustness and generalizability of our approach. We plan to integrate features from different dimensions and conduct additional comparative analyses under similar conditions. These efforts will help benchmark our results against established findings in the field and further enhance the effectiveness of our phishing detection models.

X. REFERENCES

REFERENCES

[1] Kumar, M., et al.: Machine learning models for phishing detection from tls traffic.

[2] Chiew, K.L., Yong, K.S.C., Tan, C.L.: A survey of phishing attacks: Their types, vectors and technical approaches. Expert Systems with Applications 106, 1–20 (2018)

[3] Greene, K., Steves, M., Theofanos, M.: No phishing beyond this point. Computer 51 (2018)

[4] Aung, E.S., Zan, C.T., Yamana, H.: A survey of url-based phishing detection. In: DEIM Forum (2019)

[5] Aloul, F.A.: The need for effective information security awareness. Journal of advances in information technology 3(3), 176–183 (2012)

[6] Gartner Research: Gartner survey shows phishing attacks escalated in 2007 (2007)

[7] Rao, R.S., Vaishnavi, T., Pais, A.R.: Catchphish: detection of phishing websites by inspecting urls. Journal of Ambient Intelligence and Humanized Computing 11, 813–825 (2020)

[8] Sahingoz, O.K., et al.: Machine learning based phishing detection from URLs. Expert Systems with Applications 117, 345–357 (2019)

[9] Berners-Lee, T., Masinter, L., McCahill, M.: Uniform resource locators (URL) (1738) (1994)

[10] Zhang, J., Porras, P.A., Ullrich, J.: Highly predictive blacklisting. In: USENIX Security Symposium (2008)

[11] Ma, J., et al.: Beyond blacklists: learning to detect malicious web sites from sus- picious urls. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2009)

[12] Whittaker, C., Ryner, B., Nazif, M.: Large-scale automatic classification of phishing pages (2010)

[13] Abu-Nimeh, S., et al.: A comparison of machine learning techniques for phish- ing detection. In: Proceedings of the Anti-phishing Working Groups 2nd Annual eCrime Researchers Summit (2007)

[14] L'Huillier, G., et al.: Latent semantic analysis and keyword extraction for phishing classification. In: 2010 IEEE International Conference on Intelligence and Security Informatics (2010). IEEE

[15] Marchal, S., et al.: Phishstorm: Detecting phishing with streaming analytics. IEEE Transactions on Network and Service Management 11(4), 458–471 (2014)

[16] Marchal, S., et al.: Know your phish: Novel techniques for detecting phishing sites and their targets. In: 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS) (2016). IEEE

[17] Marchal, S., Saari, K., Singh, N., Asokan, N.: Know your phish: Novel techniques for detecting phishing sites and their targets. Unknown Journal (2015). October 2015

[18] Verma, R., Dyer, K.: On the character of phishing urls: Accurate and robust statistical learning classifiers. In: Proceedings of the 5th ACM Conference on Data and Application Security and Privacy (2015)

[19] Ramanathan, V., Wechsler, H.: Phishing detection and impersonated entity dis- covery using conditional random field and latent dirichlet allocation. Computers & Security 34, 123–139 (2013)

[20] Ma, J., et al.: Identifying suspicious urls: an application of large-scale online learning. In: Proceedings of the 26th Annual International Conference on Machine Learning (2009)

[21] Mohammad, R.M., Thabtah, F., McCluskey, L.: Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications 25, 443–458 (2014)

[22] Ardi, C., Heidemann, J.: Poster: Lightweight content-based phishing detection. Tech. Rep. ISI-TR-2015-698, USC/Information Sciences Institute (2015)

[23] Wang, G., et al.: Verilogo: Proactive phishing detection via logo recognition (2011)
[24] Shahrivari, V., Darabi, M.M., Izadi, M.: Phishing detection using machine learning techniques. arXiv preprint arXiv:2009.11116 (2020)
[25] Cheng, W., Greaves, C., Warren, M.: From n-gram to skipgram to concgram. International journal of corpus linguistics 11(4), 411–433 (2006)
[26] Mahesh, B.: Machine learning algorithms-a review. International Journal of Science and Research (IJSR) 9, 381–386 (2020)
[27] Bhavitha, B.K., Rodrigues, A.P., Chiplunkar, N.N.: Comparative study of machine learning techniques in sentimental analysis. In: 2017 International Con- ference on Inventive Communication and Computational Technologies (ICICCT) (2017). IEEE
[28] Huang, A.C., Meng, S.H., Huang, T.J.: A survey on machine and deep learn- ing in semiconductor industry: methods, opportunities, and challenges. Cluster Computing 26(6), 3437–3472 (2023)
[29] Jyothish, A., Mathew, A., Vinod, P.: Effectiveness of machine learning based android malware detectors against adversarial attacks. Cluster Computing, 1–21 (2023)
[30] El Naqa, I., Murphy, M.J.: What Is Machine Learning?, pp. 3–11. Springer, Cham, Switzerland (2015)
[31] Arrieta, A.B., et al.: Explainable artificial intelligence (xai): Concepts, tax- onomies, opportunities and challenges toward responsible ai. Information fusion 58, 82–115 (2020)
[32] Nguyen, H.T.T., et al.: Evaluation of explainable artificial intelligence: Shap, lime, and cam. In: Proceedings of the FPT AI Conference (2021)
[33] Hannousse, A., Yahiouche, S.: Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applica- tions of Artificial Intelligence 104 (2021)
[34] Sameen, M., Han, K., Hwang, S.O.: Phishhaven—an efficient real-time ai phishing urls detection system. IEEE Access 8, 83425–83443 (2020)
[35] Marchal, S., et al.: Phishstorm: Detecting phishing with streaming analytics. IEEE Transactions on Network and Service Management 11(4), 458–471 (2014)
[36] Saxe, J., Berlin, K.: expose: A character-level convolutional neural network with embeddings for detecting malicious urls, file paths and registry keys. arXiv preprint arXiv:1702.08568 (2017)
[37] Rao, R.S., Vaishnavi, T., Pais, A.R.: Catchphish: detection of phishing websites by inspecting urls. Journal of Ambient Intelligence and Humanized Computing 11, 813–825 (2020)
[38] Jain, A.K., Gupta, B.B.: Phish-safe: Url features-based phishing detection system using machine learning. In: Cyber Security: Proceedings of CSI 2015. Springer, Kurukshetra 136119, Haryana, India (2018)
[39] Chen, W., Zhang, W., Su, Y.: Phishing detection research based on lstm recur- rent neural network. In: Data Science: 4th International Conference of Pioneering Computer Scientists, Engineers and Educators, ICPCSEE 2018, Zhengzhou, China, September 21-23, 2018, Proceedings, Part I (2018)