Building a Multi-class Prediction App for Malicious URLs

Vijayaraj Sundaram[1][0000-0002-9556-0865], Shinu Abhi[2][0000-0002-3318-7479]

and Rashmi Agarwal[3][0000-0003-1778-7519]

[1,2,3] REVA Academy for Corporate Excellence, REVA University, Bangalore, India

[1]vijayraj.cs04@race.reva.edu.in
[2]shinuabhi@reva.edu.in
[3]rashmi.agarwal@reva.edu.in

Abstract. The page that houses a malicious snippet that could misuse a user's computing resources, steal confidential data, or carry out other forms of assaults is known as a malicious host URL. They are generally distributed across the world wide web under various usage categories like spam, malware, phishing, etc. Although numerous methods or fixes (to identify URLs) have been developed in recent years, still cyberattacks continue to occur.

This study contributes towards implementing three tiers of the system for detection and protection from harmful URLs. The first tier focuses on evaluating the performance of discriminative features in model creation. Discriminative features are derived from URL details and "*Whois*" webpage information that helps in improving detection performance with less latency and low computational complexity. The influence of feature variation on Parametric (neural network) and non-parametric classifier detection results are assessed to narrow down to the most prominent features to be adapted in the best model for the task of identifying URLs with multi-categorization. The study reveals that non-parametric ensemble models like Light GBM, XGBoost, and Random Forest performed well with a detection accuracy of over 95%, which facilitated building a real-time detection system and differentiating multiple attack types (such as Malware, Phishing, and spam).

The second tier focuses on validation with a global database to know, if entered URL is reported as suspicious by various detection engines already. If not, it enables the user in updating the global database with URL details that are new and not reported yet. Finally, the two modules are integrated to create a web application using *Streamlit* that provides full system protection against malicious URLs.

Keywords: Multiclass classification, Malicious URLs, Ensemble Learning, nonparametric models, Prediction.

1 Introduction

During the Covid phase, the predominant of our daily work transformed towards online and online platforms. This behavior continued even after covid period due to the convenience it brings to end users fulfilling their individual needs with a click away. As more people, organizations, and services turn to the Internet to meet their needs, it has turned into a breeding ground for various forms of fraudulent activities [1]. Malicious attacks including phishing, spamming, and malware infection have all been carried out by adversaries using the Internet as a delivery system [2].

It can be challenging for Internet users to stay on top of the most recent dangers, frauds, viruses, malware, and other forms of malevolent activity. Cybercriminals take benefit of this situation by employing phishing webpages. Spyware, Malware, and a virus are implanted in those websites [1]. Online scams have become more prevalent, mostly as a result of malicious Uniform Resource Locators (URLs) that lead to dangerous websites. As per CISCO's report on cybersecurity threat trends for 2021, approx. 85 percent of firms have one employee who clicks on a phishing link which led to 90% of data breaches [2]. Users clicking on bad URLs are reported as the second most expensive cause of data breaches. Compromised credentials ranked 5th costliest cause of data breach in an organization [3].

Different methods are suggested to identify malevolent weblinks and detrimental content by extracting discriminative rules or features from their URLs [4][5]. Most of the methods depend on the manual derivation of the URL features [6][7] while other solutions used deep learning techniques to mechanize feature creation [8][9]. Attackers bypass security countermeasures easily by using deceptive tactics. The URL-based features may not be sufficient for effective representation because they can be dynamically altered and are open to manipulation by attackers. Therefore, elements that are outside the control of attackers can increase detection accuracy and brings down false alarms.

Domain attributes are one such element that can enrich statical URL features to improve the precision of detection and a lower false alert rate. As deep learning and ensemble algorithms get powerful in reading complex patterns and getting profound insights, it brings a need to explore the adaptability of those algorithms to apply in multi-classification and detection of malevolent URLs with suitable features generated for it.

This work explores the impact of URL and domain features influence on multi-class detection accuracy of Deep Learning (DL) and ensemble models. Having the right classifier in conjunction with global knowledge bases from the web brings the double benefit of detection and protection to end users.

The four primary objectives of this study are to evaluate the performance of discriminative features derived from URL details while building a multiclassification

model for detecting URLs. Explore the influence of domain feature variation on the detection result of parametric (Artificial Neural Network) and nonparametric (Ensemble) models. Build integration with a third-party tool (Virus Total) for parallel validation with various detection engines and finally build a single go-to concept web portal focusing on simplified user input for detecting malicious URLs with a basic web User Interface (UI).

The rest of the manuscript is structured as follows. Related literature works are reviewed in Section II. Section III explains the detection system design methodology, data preprocessing, feature generation methods, and algorithm selection process for building multi-category URL classification. Section IV explains feature visualization, performance on model detection, and development of a web predictor app. Section V describes the experimental outcome and discusses in detail the results. Section 6 concludes the paper and shares future work.

2 Literature Review

The identification of malicious URLs has drawn a lot of attention from researchers in recent years. The most alluring research techniques among these many techniques rely on machine learning. As a result, it is gradually becoming a common practice to use machine learning algorithms in the detection activities of fraudulent domain names [10][11][12][13].

The machine learning features are gleaned from a variety of sources, including Domain Name System (DNS) traffic, *whois* data, search engine results, and online content. The authors of [14] used a mix of google features along with static URL and host features to build ensemble models utilizing Random Forest and Multilayer Perceptron (MLP) algorithms for binary classification. The deep learning and ensemble algorithms getting powerful in reading complex patterns and getting profound insights which are explored by authors of [15][16][9][17].

Web content analysis involves visiting the page and performing content analysis for detecting if the website is malicious or not. It brings a huge performance bottleneck, and it is too late to prevent an attack since URL is already opened for study. Hence methods focusing on filtering in advance explored by authors of [18] to identify highly suspect malevolent URLs in massive networks to ease the computational load on back-end systems for detecting bad URLs based on content analysis. Their solution achieved a high recall rate by utilizing simple features.

I. Qabajeh et al. focused on comparing conventional and automated anti-phishing techniques in their work. In addition to highlighting the differences, similarities, and good and bad elements of these approaches from the users' and performance perspectives, they presented methods to counteract phishing by both conventional and intelligent means[19]. Author R. S. Rao et al. built a desktop program named PhishShield that focuses on phishing page URL and Website Content in [20]. PhishShield accepts

a URL as an input and returns information about whether it is a real or phishing website.

The heuristics for phishing detection include copyright content, title content, zero links in the HyperText Markup Language (HTML) body, etc. Malicious URLs in the email are detected through the reduced feature set method since hackers pose as a legitimate sender purporting to be from a legitimate institution and ask the recipient for his personal information. A training dataset of harmful phishing URLs and trustworthy URLs is taken for the classifier validation [21].

The authors proposed a machine learning-based approach using a logistic regression algorithm to identify adversaries from the URL input data in the study [22]. Their framework is evaluated further against conventional malicious URL models to measure performance in terms of time taken for training and testing the algorithms. Global Vector for Word Representation (GloVe) is used as a feature to enhance the accuracy of the ML model in detecting malevolent URLs has been proposed by [23]. Features used in their work include work vector representation which is obtained from GloVe alongside statistical cues and running n-gram on blacklists words. Thus, they achieved a prediction accuracy of 89 percent using the ANN algorithm along with GloVe features.

Hevapathige, Asela, et al. work centered on developing an ML classifier model using the Super Learner ensemble to categorize dangerous URLs in [24]. Using the dataset, they built both a binary classification model and multiclassification model which delivered an accuracy rate of 95% and a prediction rate of 96% as against multi classifiers which gave accuracy and prediction rate of 94 % and 96% respectively. A Multilayer ML model is proposed for detecting malicious URL by authors in [25]. Four filters or layers are constructed in the work composing black and whitelist, Naïve Bayesian, Classification and Regression Trees (CART) Decision Tree (DT), and Support Vector Machine (SVM) classifier. This improved URL detection in terms of accuracy in their work.

The authors built a phishing webpage using FLASK for detecting harmful URLs by exploring various ML classifiers using three approaches for text feature extraction, count vectorizer, hashing Inverse Document Frequency (IDF) vectorizer, and built models with Decision Tree, K-Nearest Neighbors (KNN), Random Forest, and Logistic Regression in [26]. An accuracy of 97.5 % was achieved from the model which used a hash vectorizer together with a random forest algorithm. The outcome of this work also covered building a web application using Flask to check if the entered URL is malevolent or not.

Authors T. Alsmadi, N. Alqudah et al. studied various methods used for malware detection and explored heuristic detection techniques for malware identification in their study [27]. The work reviewed extensively various features used in methods like Application Programming Interface (API) Calls, N-Grams, OpCodes, etc. and lists down in detail the disadvantages and advantages of features applied in the heuristic

methods. Verma, Manisha et al. [28] present a comprehensive study and fundamental understanding of non-benevolent URL detection techniques under machine learning tasks. Authors sort and evaluate the contributions of literary works that take into account the various aspects of the issue such as feature representation, algorithm design, etc.

In conclusion, a variety of approaches have been suggested for identifying fraudulent URLs using binary classification i.e., good, or bad. For categorization, the majority of these systems use supervised-based machine learning methods. Investigations have also been done into the deep learning strategy. Very few works are done to the best of knowledge on multi-class classification of URLs.

A threat actor can cause harmful cyber-attacks by sending Malicious URLs for different purposes under various categories. To evaluate and find these URLs, one must be able to recognize them. Recognition of attack kinds is important since it enables us to respond appropriately and take relevant countermeasures against threats by understanding their nature. For instance, malware infection needs immediate response whereas spamming may be overlooked conveniently. Hence, this study focuses on the top 3 frequent and prominent attacks namely malware, spam, and phishing. The next section offers a thorough explanation of the suggested methodology.

3 Proposed Methodology

As described in much of the literature, experimental studies on certain lexical and host features including link structures that are highly discriminative in nature are not only detecting malicious URLs but also identifying various attack kinds or types (Malware, Phishing, and Spam). Hence this study adapts a large set of discriminative URL features related to link structures, textual patterns, domain information, and URL content composition. Many of these features' effectiveness needs to be explored for a good understanding of classifying URLs under various types in the context of algorithm performance.

The Artificial Neural Network (ANN) classifier which is also a parametric and non-parametric ensemble classifiers are taken into consideration in this work to examine and choose a suitable detection model to be incorporated into a web application. This is done to have better classification and to understand classifier's ability to extract effectiveness patterns from textual-based features. Fig. 1 illustrates the proposed methodology for building a detection system of a web application with the integration of multiple modules.

To the best of knowledge, the novelty in this methodology is application of DL and ensemble classifiers and the use of both lexical and host features for the multi-classification of URLs. Using the Python *Streamlit* framework for building web ap-

plications and integration of Virus Total in the prediction results adds uniqueness to the solution.

3.1 Data Preprocessing and Feature Engineering

Various steps are involved in the design phase starting with data collection and preprocessing. Real-world data contains noise, errors, partial information, missing values, and other forms of corruption. Properties of the data are key for an ML model to produce accurate and exact predictions. For this study, a dataset of total of 1,06,230 URLs with various attack types of Malware, Phishing, spam, and Benign is collected from Kaggle [30]. Dataset consists of 27424 Malware URLs, 26773 Phishing URLs, 25889 Spam URLs, and 26144 benign URLs. After exploring the data through data visuals and summaries, the next step involves is to add more information to variables for increasing their significance and make them more meaningful.

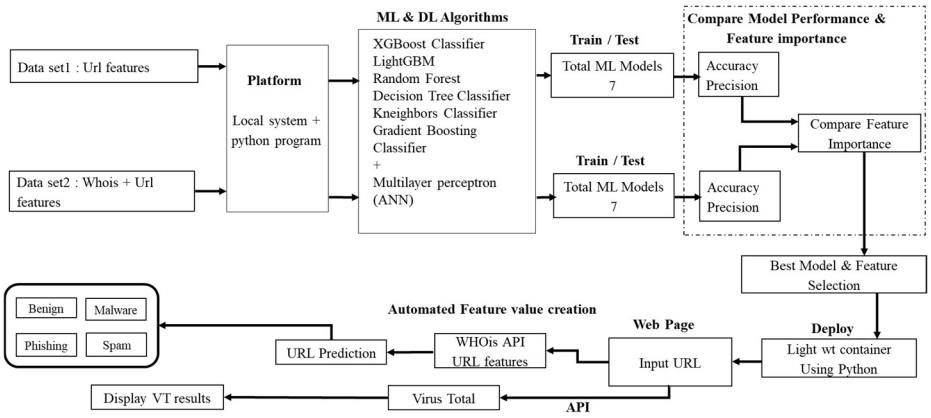

Fig. 1. Proposed Detection system design

Two datasets with the difference in features are created for the study. Static lexical features are created from each URL by applying text preprocessing techniques to the raw data. Tokenization, word embedding, and term frequency methods are used in the process of feature engineering steps to generate Dataset1 with 27 features as shown in Fig. 2.

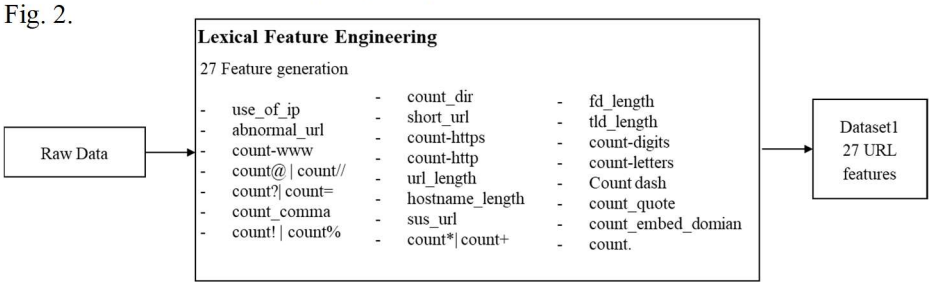

Fig. 2. Dataset1 creation with URL features

An in-house developed python program is used to collect domain registration details of each URL from the *whois* portal [31]. Information like domain registration date, expiration date, and registered nation are used to create two features relating to domain age and host country. These 2 *whois* features are added to dataset 1 to create dataset 2 illustrated in Fig. 3.

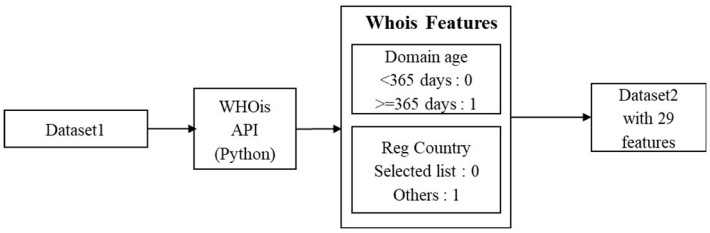

Fig. 3. Dataset2 creation adding domain features

3.2 Deep Learning and Ensemble Algorithm Selection

Classifiers are models of data that have a categorical response. Different classifier algorithms hold varying presumptions or biases regarding the structure of the function and its learnability. Although they can substantially ease learning, assumptions also have a limit on what can be learned. Parametric algorithms are those that reduce the event to a recognized form. The benefits of Parametric Algorithms are it follows simpler procedures, have quick learning speed, and can work well with less training data. But it comes with constraints in terms of handling limited complexity and can lead to poor fit.

Whereas Nonparametric algorithms are the one that does not make up any firm theories about the structure of the mapping function. They are allowed to discover any functional pattern from the training dataset because they are not making any assumptions. These classifier algorithms are able to fit a wide variety of functional forms and can lead to prediction models with greater performance. But it also comes with limitations like the demand of more data volume for training purposes and comes with the risk of overfitting the training data.

In this study, a combination of parametric and non-parametric algorithms is considered to derive a suitable model for multi-class URL detection. Multilayer perceptron (MLP) is taken for study under the parametric algorithm category which is also eventually the Artificial Neural Network (ANN) algorithm. Decision Tree (DT) classifier, K-Nearest Neighbor (KNN) Classifier, Gradient Boosting (GB) Classifier, Random Forest (RF) Classifier, Light Gradient Boosting Machine (LGBM) Classifier, and Extreme Gradient Boosting (XGBoost) Classifiers are taken for an experiment under nonparametric category. XGBoost, LGBM, GB and RF are also eventually ensemble algorithms.

3.3 Experimental Procedures and Model Performance Validation

Two datasets with differentiating features are taken for study to understand the discriminative nature of lexical and host features applicability in detecting malicious URLs and also to identify various attack types efficiently. This enabled, the analysis of the influence of feature variation on the detection result of parametric and non-parametric models that suit to be considered for building URL predictor web application. Experiments run with 7 algorithms on each dataset, to understand the discriminative nature of features. The high-level workflow of the experiment is shown in Fig. 4.

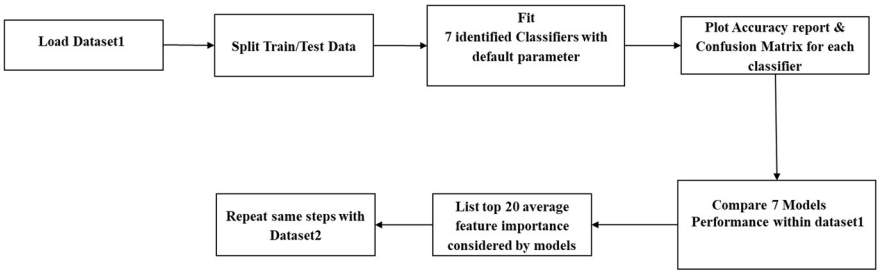

Fig. 4. High-level workflow of model experiments

The model's prediction accuracy of classes in runtime and in training are considered as model performance. It makes it noticeable for the model to show its effective exploitation of the dataset's scoring. A total of four performance, assessment metrics are considered for model validations as Accuracy rate (1), Precision (2), F1 Value (3), and Recall rate (4). False Positive, True Positive, False Negative, and True Negative are used to indicate detection findings which are abbreviated as FP, TP, FN, and TN respectively.

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

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$F1\ score = \frac{TP}{TP\ \ \frac{1}{2}(FP+FN)} \tag{3}$$

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

Dataset is split into ratio 80:20 for train and test purposes. Two experiments are done using two datasets employing 7 algorithms each time. A total of 14 models are created in the study to compare the performance metrics of models. Also, the top 20 average feature importance considered by models are measured to check the list of features

having the maximum impact on model performance. In experiment one with dataset1 carrying only URL lexical features, XGBoost classifiers and Random Classifier gave maximum prediction accuracy and precision rate of 95% each.

The impact of adding the *whois* feature in dataset2 is reflected in enabling additional model LGBM to provide maximum performance together with XGBoost and RF classifiers. In addition, the performance of the ANN MLP classifier increased by 1% but not equal to or exceeding the top 3 models mentioned above. It provides the inference that 29 features extracted from domain and URL links maximizes model performance accuracies.

3.4 Build Web Predictor for URL categorization

Web application enable users to have the live user interface for the purpose it is built. In this study, a web predictor is designed to host deployed model along with 3 other modules which are integrated for prediction purposes. For generating features from user input URL, the *Whois* connector module and URL feature builder modules are developed. High-level workflow design of the modules is illustrated in Fig. 5 and Fig. 6.

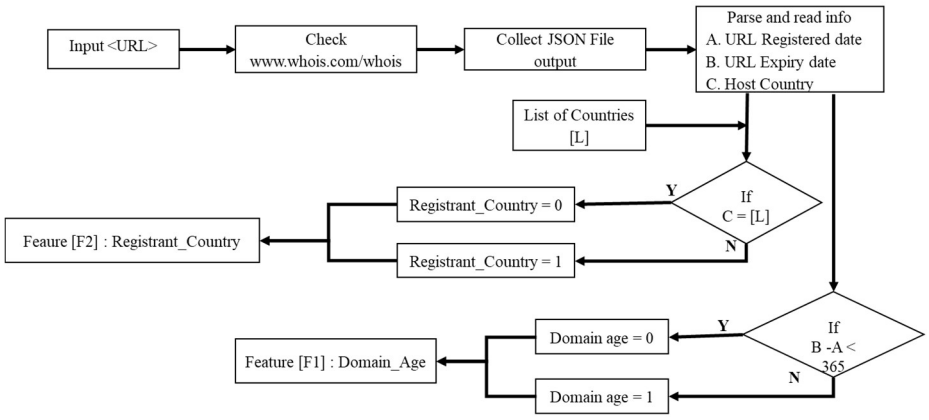

Fig. 5. *whois* feature builder module workflow

Fig. 6. URL feature builder module workflow

Virus Total (VT) web page provides free service that scans files and URLs for malware such as viruses, worms, and trojan horses. They collaborate with various antivirus organizations and researchers for utilizing scanners to provide safe internet for all users. In this work, a third module is built using a python program to ingest the URL

entered in the input field of the web app and the value is passed to Virus Total for a check, consuming free tier API. The received output value is displayed in the web app. This provides a second layer of protection to end users in our solution. The workflow of this module is depicted in Fig. 7.

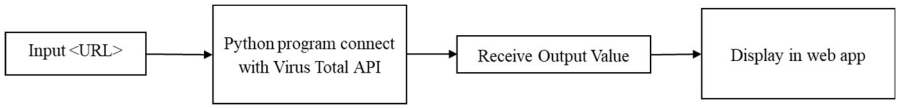

Fig. 7. Virus Total module workflow

An open-source Python framework called *Streamlit* is used to create predictor web application by integrating all modules along with a machine learning model as shown in Fig. 8. Using *Streamlit*, one can quickly design and launch web applications. *Streamlit* can create apps in the same manner that one creates with Python code. Working on the interactive cycle of coding and watching outcomes on the web app is made simple by *Streamlit*.

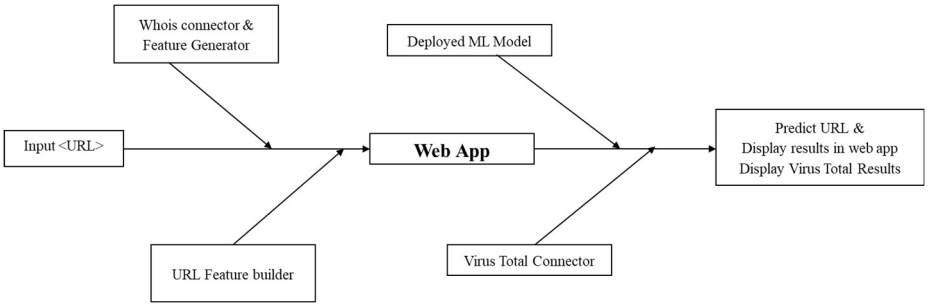

Fig. 8. URL Predictor Web app workflow and module integrations

3.5 Experimental Settings

The experimental setup is created using computer configuration: Windows 11, 32GB DDR4 3200MHz of memory, CPU AMD Ryzen™ 5 4600H (3.00 GHz base frequency, up to Max. Boost Clock 4.0GHz, 8 MB cache, 6 cores, 12 Threads), 256 GB PCIe® NVMe™ M.2 SSD + 1 TB HDD of hard disk; Python version:3.9.12; Anaconda Navigator, Jupyter IDLE, Scikit Learn, *Streamlit*.

4 Visualization, Performance Evaluation and Results

4.1 Datasets

Comprehensive details on the dataset are provided in this area. A publicly accessible URL dataset in Kaggle is used for this investigation. Using data visualization tools

and exploratory data analysis approaches, it is understood that datasets have an equal distribution of benign, malware, phishing, and spam URLs as shared in Figure No. 9 and Table 1.

Top5 records of raw data are shown in below Fig. 10. Using the raw data, two domain features are generated from *whois* registered information. Fig.11 shares that new URLs with an age of less than one year are used predominantly for Malware and phishing attacks compared to benign and spam URLs.

Post-generating URL lexical features, it is inferred that a particular URL length is used for creating malware and phishing websites which is illustrated in graph Fig. 12.

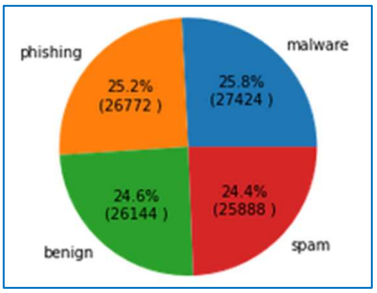

Fig. 9. URL attack type composition in raw data.

Table 1. Raw data details

Label	No. of URLs
Benign	26144
Malware	27424
Phishing	26773
Spam	25889
Total URLs: 1,06,230	

Out[7]:

	url	type
0	http://www.keve-kiserdo.hu/portal/index.php?op...	spam
1	www.agdealer.com/keystone	phishing
2	sbcspa.tripod.com/sbcs/	phishing
3	hockeydraft.ca/players/profile.aspx?id=4132&na...	benign
4	210.37.11.238/jm32/includes/site/config.bin	malware

Fig. 10. Top 5 records of raw data.

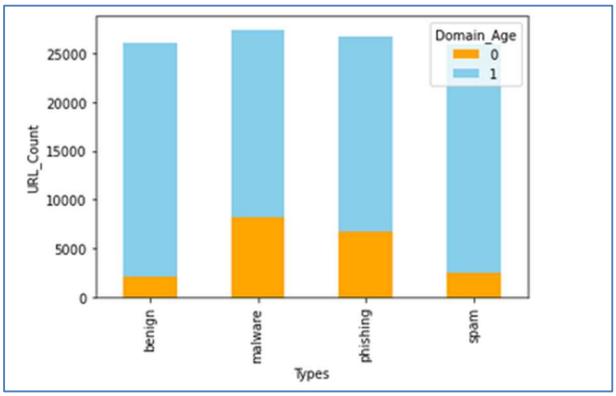

Fig. 11. URLs with age less than365 days.

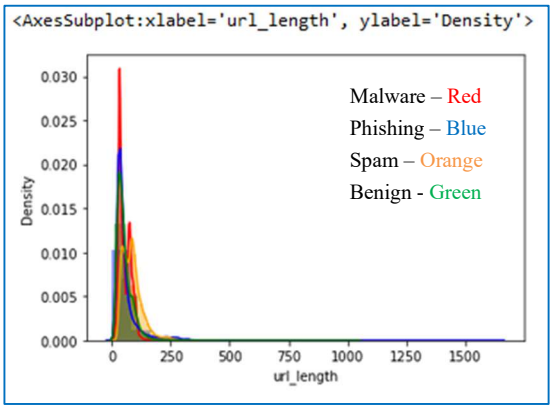

Fig. 12. URLs length as per Labels

4.2 Experiment with Lexical features (Dataset1)

Total of 7 models are constructed using 27 URL features from Dataset 1. RF, DT, KNN, LGBM, Gradient Boost, XGBoost, and MLP classifiers are used with the default parameters as it is. Fig. 13 illustrates the accuracy of all Models and Table 2 shares the performance metrics of different classifiers. Nonparametric classifiers such as XGBoost and Random Forest gave maximum accuracy and precision of 95% each.

Fig. 14 depicts top 20 features which influenced model performance when using Dataset1. First directory length, hostname length, URL length, and directory counts are the top 4 features that influenced model performances. Fig. 15 – 21 shows the confusion matrix of various models.

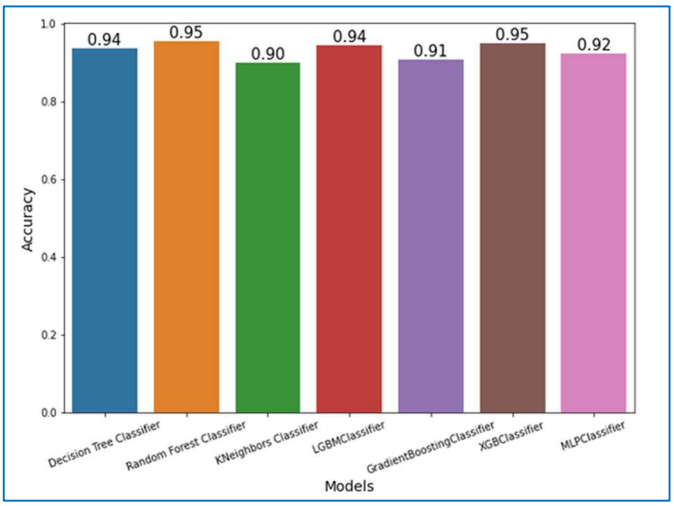

Fig. 13. Accuracy of models when using Dataset 1.

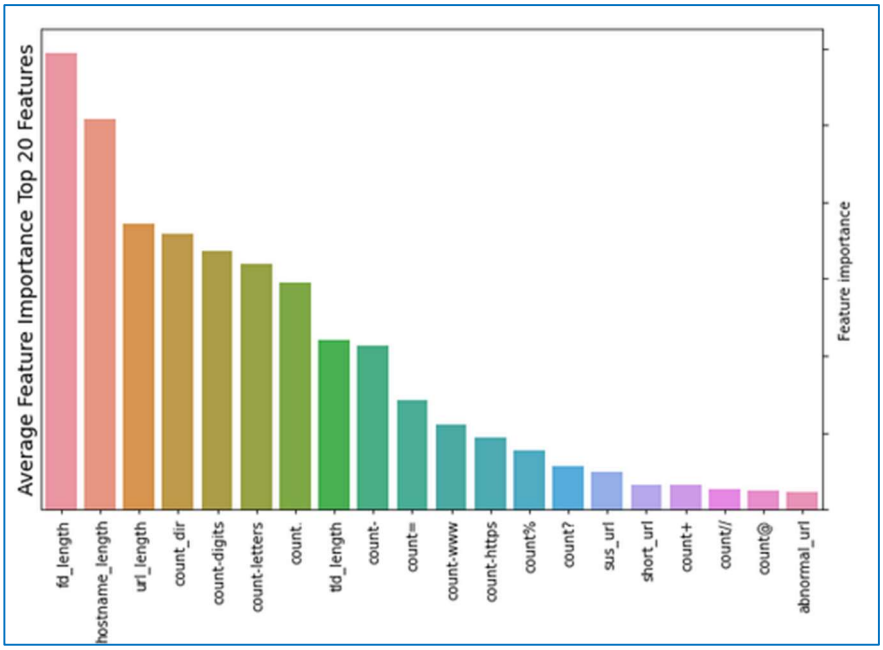

Fig. 14. Top 10 features influencing model performance when using URL features (Dataset1)

Table 3. Performance metrics of classifiers when using URL features (Dataset1).

Classifier Type	Classifier Name	Accuracy	Precision	Recall	F1
Non-parametric	DT	94%	90%	87%	89%
	RF	95%	93%	90%	91%
	KNN	90%	88%	80%	83%
	LGBM	94%	92%	88%	90%
	GB	91%	89%	81%	85%
	XGBoost	95%	93%	95%	94%
Parametric	MLP	92%	90%	82%	87%

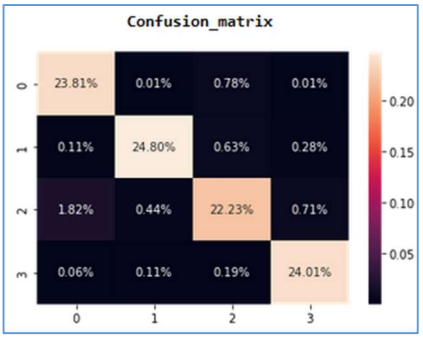

Fig. 15. XGBoost Confusion matrix

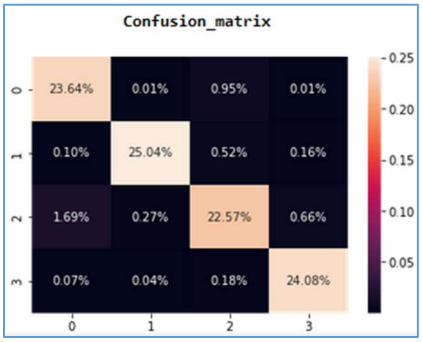

Fig. 16. RF Confusion matrix

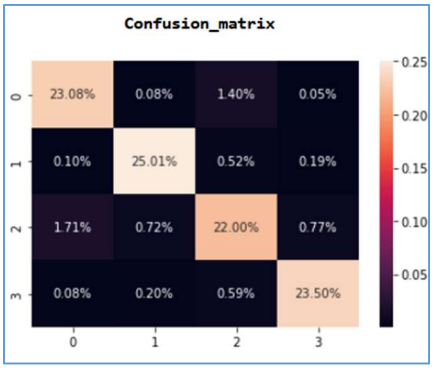

Fig. 17. DT Confusion matrix

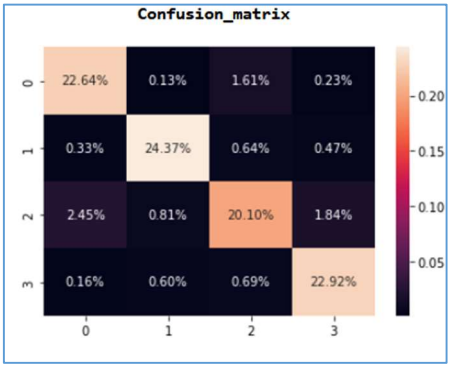

Fig. 18. KNN Confusion matrix

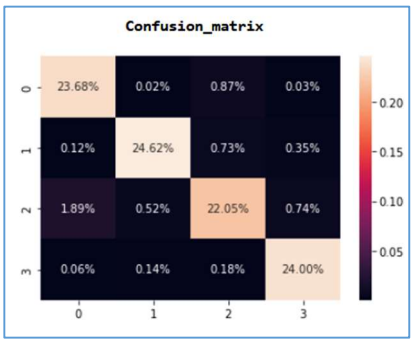

Fig. 19. LGBM Confusion matrix

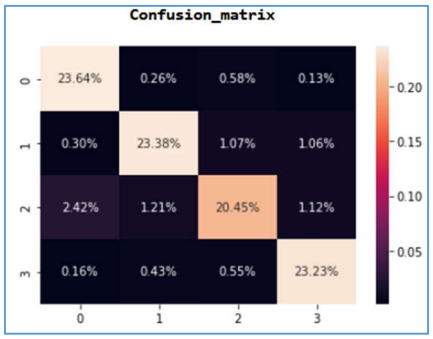

Fig. 20. GB Confusion matrix

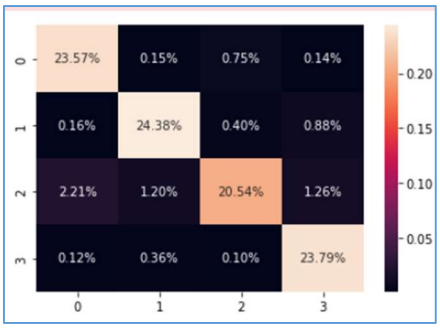

Fig. 21. MLP Confusion matrix

4.3 Experiment with Domain and Lexical features (Dataset2)

The impact of adding two domain features along with 27 URL features are studied by employing the same list of 7 classifiers with default parameters by building seven experiment models. It resulted in increasing more model's prediction accuracy compared to the experiment with Dataset 1. Again, nonparametric models like LGBM, XGBoost, and RF performed well with maximum accuracy. The parametric model i.e., MLP's accuracy as well increased by 1% but not matching the maximum efficiency of other models. Fig. 22 illustrates the accuracy of all Models and Table 4 shares the performance metrics of different classifiers. Fig. 23 to 29 shows the confusion matrix of all models.

Fig. 30 depicts top 20 features which influenced model performance when using Dataset2 i.e., with addition of *whois* features. It clearly illustrates that Domain Age and Registrant country has a relatively good impact on the model performance.

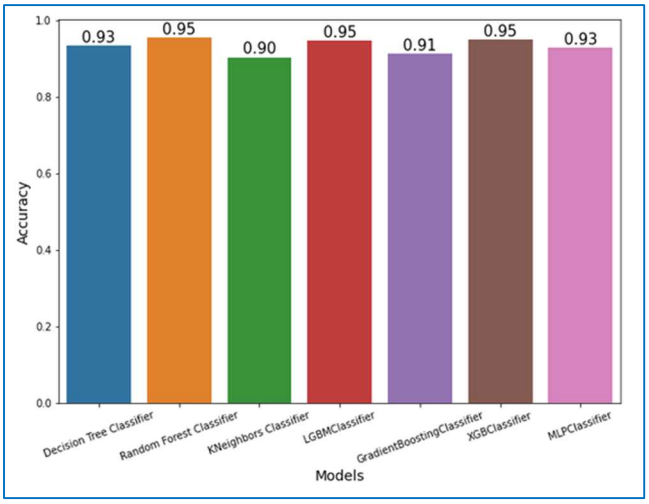

Fig. 22. Accuracy of models when using Dataset 2.

Table 4. Performance metrics of classifiers when adding *whois* features (Dataset2).

Classifier Type	Classifier Name	Accuracy	Precision	Recall	F1
Non-parametric	DT	93%	93%	87%	88%
	RF	95%	93%	88%	92%
	KNN	90%	88%	80%	84%
	LGBM	95%	96%	88%	91%
	GB	91%	90%	82%	86%
	XGBoost	95%	95%	96%	97%
Parametric	MLP	93%	93%	90%	89%

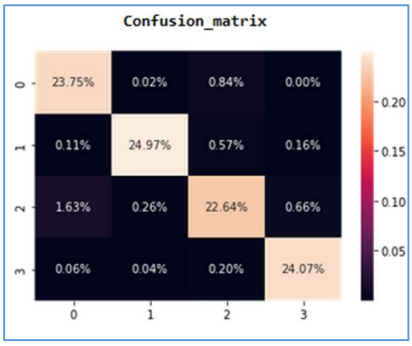

Fig. 23. RF Confusion matrix

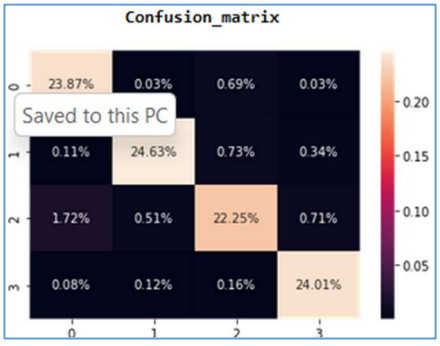

Fig. 24. LGBM Confusion matrix

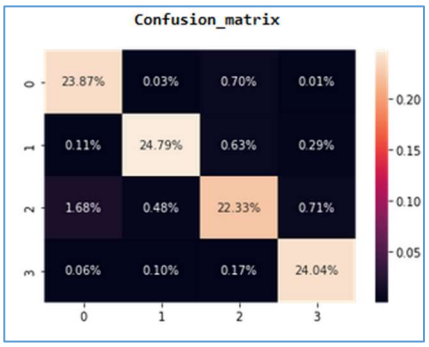

Fig. 25. XGBoost Confusion matrix

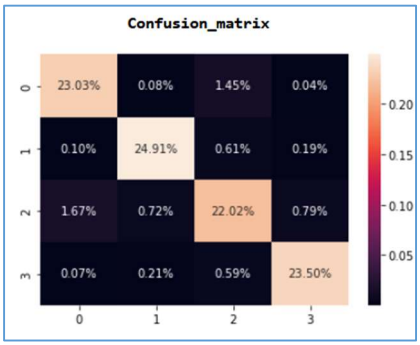

Fig. 26. DT Confusion matrix

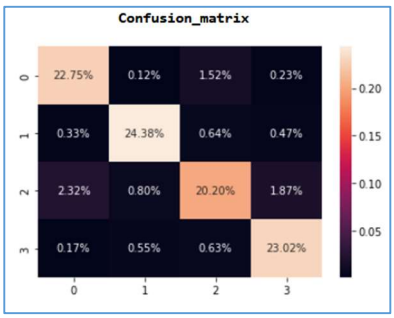

Fig. 27. KNN Confusion matrix

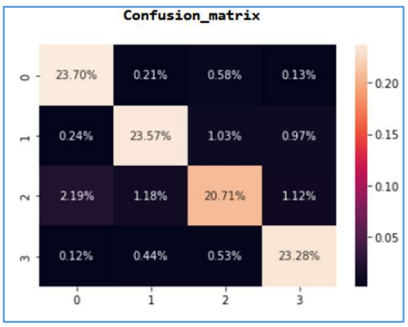

Fig. 28. GB Confusion matrix

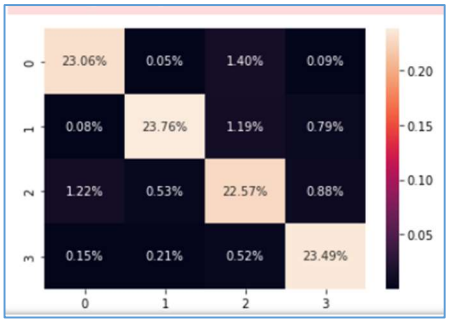

Fig. 29. MLP Confusion matrix

4.4 Web application for Prediction

It is observed in the experiment that domain features bring value and increase efficiency in model prediction. Since nonparametric classifiers are performing well for multi-class classification of detecting URLs, the XGBoost model is selected for deployment along with *whois* and lexical URL features.

Streamlit is an open-source Python framework that makes it simple to develop and distribute personalized web applications for data science and machine learning. Since they reduce our technical debt by over 80%, using *Streamlit* for Machine Learning applications has shown to be more cost-effective in terms of maintenance. Hence the final model is deployed using a lightweight container and a web application is built using *Streamlit* for URL prediction and multi categorization purposes.

Streamlit web app is compatible with all browsers (Google Chrome, Firefox, Microsoft Edge, and Safari) in systems with operating systems like Windows, Unix, and macOS. Prerequisites for a web server hosting environment are Python 3.7 – 3.10 and PIP.

Virus Total connector is developed and integrated as a background service along with two feature builder modules that use domain and URL lexical information for dynamic feature creation from input URL. Fig. 31 shows the web app GUI.

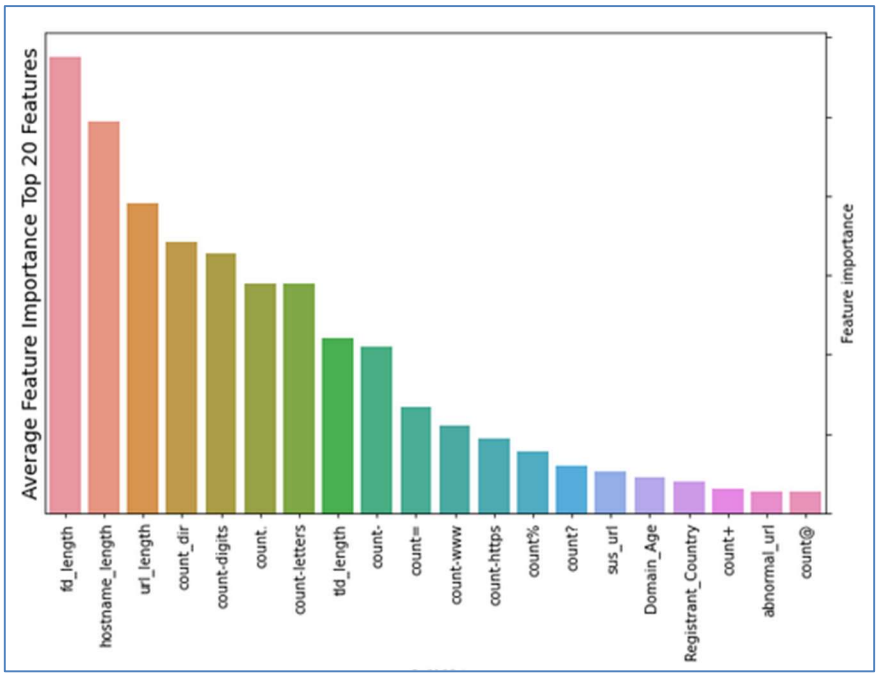

Fig. 30. Top 20 features influencing model performance when adding *whois* feature (Dataset2)

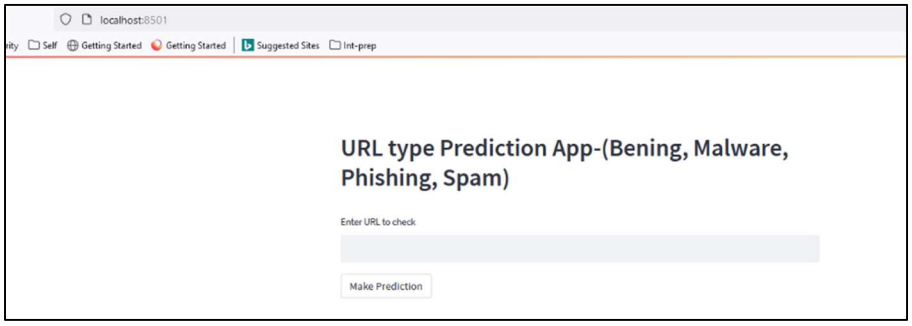

Fig. 31. URL Web Predictor Web app GUI

5 Analysis and Results

It is important to note that both the Nonparametric (ensemble) classifier LGBM and Parametric ANN classifier MLP performance increased as the data is enriched with more features. Nonparametric ensemble classifiers like XGBoost and Random Forest models demonstrate performance consistency staying high with maximum prediction accuracy for multi-classification of URLs compared to other models. A summary of the model's performance and comparison across two feature sets are illustrated with a prediction accuracy trend in Table 4. A comparative analysis of performance indicators with similar works referred to in the literature review is illustrated in Table 6. With respect to feature importance considered by models, first directory, hostname length, URL length, directory, and digit counts having maximum impact on model performances. A Simple webpage seeking input as URL, predicting URL type (Benign, Malware, Phishing, Spam), and displaying Virus Total results worked fine in the same webpage as shown in Fig. 32.

Table 5. Models' performance summary and comparison across two datasets.

	Prediction Accuracy		
Classifier Name	**Dataset1 URL features**	**Dataset2 URL + Whois Features**	**Trend**
RF	0.95	0.95	→
LGBM	0.94	0.95	↑
KNN	0.9	0.9	→
GB	0.91	0.91	→
DT	0.94	0.93	↓
XGBoost	0.95	0.95	→
MLP	0.92	0.93	↑

Table 6. Comparative analysis of performance indicators with previous works.

Classifier Name	Accuracy	Precision	Recall	F1
Asela et al. [24]	82.37%	89.78%	58.22%	70.64%
Rohit et al. [23]	89.00%	88.00%	88.00%	92.00%
Chiramdasu et al. [22]	91.00%	62.00%	85.00%	90.00%
Eduardo et al. [9]	93.47%	93.63%	93.28%	93.46%
Chen et al. [8]	85.99%	87.19%	84.66%	85.01%
Zhao et al. [5]	90.31%	90.03%	90.00%	89.00%
Proposed work	**95.02%**	**95.04%**	**96.10%**	**96.99%**

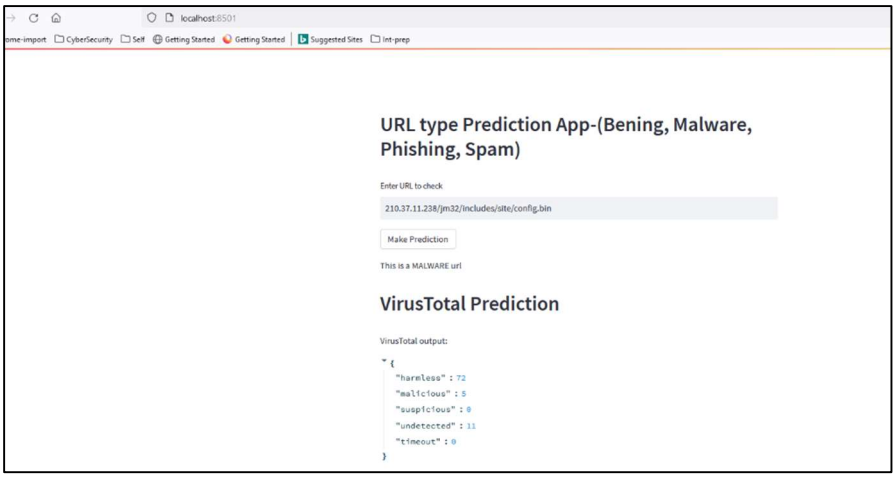

Fig. 32. URL Web Predictor Output results

6 Conclusion

This study proposed a unique multi-class detection App for identifying malicious URLs and classifying the same into different attack categories. The model inbuilt into the app was designed and developed based on domain data obtained from the web and using acumen extracted from URL structure. Feature engineering is an important step in malicious URL detection. Each model focuses on an integrated aspect of high dimensional features, leveraging the merits of nonlinear, linear, supervised, parametric and nonparametric models.

The conducted experiments using 106230 instances of web URLs demonstrated that the proposed feature engineering methods are effective and can considerably

improve the performances of few types of classifiers in identifying malicious URLs and detecting attack types. We applied six nonparametric classifiers and one parametric artificial neural network algorithm using two datasets. Total of 14 experiments was conducted and investigated. Out of all scenarios, non-parametric classifiers XGBoost, LGBM, and Random Forest models achieved the highest accuracy rate of 95% for URL prediction and XGBoost is used to build multiclass prediction module due to its consistency with high performance in varying features scenario.

This work also focused on developing another module, which validates input URL with a global database (Virus Total) if entered weblink is reported as suspicious or clean by various virus detection engines. This would provide an option to end users and it facilitates them in updating the global database with URL details that are new and not reported as malicious yet. Finally, the two modules are integrated by creating a web app using *Streamlit* and the final application provides full system protection against malicious URLs. This multiclass problem can be considered from the perspective of multilabel learning with little finetune in the methodology keeping class as malicious with labels as Malware, Phishing, and Spam.

Detecting malicious URLs needs continuous data collection, feature selection, and extraction with routine model training. Observation from the study is that DNS feature inclusion and construction of additional features from *whois* information can also be explored as a prominent future study. The predictor app which is built in the study can be further developed for production use and can be utilized as a complementary tool in existing or new anti-phishing, anti-spam, and anti-malware discovery platforms.

References

[1] J. Acharya, A. Chuadhary, A. Chhabria, and S. Jangale, “Detecting malware, malicious URLs and virus using machine learning and signature matching,” *2021 2nd Int. Conf. Emerg. Technol. INCET 2021*, pp. 1–5, 2021, doi: 10.1109/INCET51464.2021.9456440.

[2] Maddie Rosenthal, “Must-Know Phishing Statistics: Updated 2022.” https://www.tessian.com/blog/phishing-statistics-2020/#phishing-by-country

[3] C. Public, “Cyber security threat trends :,” 2021.

[4] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, “Machine learning based phishing detection from URLs,” *Expert Syst. Appl.*, vol. 117, pp. 345–357, Mar. 2019, doi: 10.1016/j.eswa.2018.09.029.

[5] H. Zhao, Z. Chen, and R. Yan, “Malicious Domain Names Detection Algorithm Based on Statistical Features of URLs,” *2022 IEEE 25th Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2022*, pp. 11–16, 2022, doi: 10.1109/CSCWD54268.2022.9776264.

[6] R. George, R. Jalal, R. M. Raju, S. S. Sunny, and M. Hari, “High responsive

plug-in for malicious URL detection," *Proc. Int. Conf. Trends Electron. Informatics, ICOEI 2019*, no. Icoei, pp. 357–359, 2019, doi: 10.1109/ICOEI.2019.8862664.

[7] T. Li, G. Kou, and Y. Peng, "Improving malicious URLs detection via feature engineering: Linear and nonlinear space transformation methods," *Inf. Syst.*, vol. 91, p. 101494, 2020, doi: 10.1016/j.is.2020.101494.

[8] Y. Chen, Y. Zhou, Q. Dong, and Q. Li, "A Malicious URL Detection Method Based on CNN," *2020 IEEE Conf. Telecommun. Opt. Comput. Sci. TOCS 2020*, pp. 23–28, 2020, doi: 10.1109/TOCS50858.2020.9339761.

[9] Anti-Phishing Working Group. and Institute of Electrical and Electronics Engineers, *Classifying Phishing URLs Using Recurrent Neural Networks*.

[10] H. Choi, B. B. Zhu, and H. Lee, "Detecting malicious web links and identifying their attack types," *WebApps*, no. July 2014, p. 11, 2011, [Online]. Available: http://dl.acm.org/citation.cfm?id=2002168.2002179

[11] K. Ramesh, M. A. Bennet, J. Veerappan, and P. N. Renjith, "Performance Metric System for Malicious URL Data using Revised Random Forest Algorithm," *Proc. - 5th Int. Conf. Comput. Methodol. Commun. ICCMC 2021*, no. Iccmc, pp. 1188–1191, 2021, doi: 10.1109/ICCMC51019.2021.9418480.

[12] Shantanu, B. Janet, and R. Joshua Arul Kumar, "Malicious URL Detection: A Comparative Study," *Proc. - Int. Conf. Artif. Intell. Smart Syst. ICAIS 2021*, pp. 1147–1151, 2021, doi: 10.1109/ICAIS50930.2021.9396014.

[13] W. Zhang, H. Ren, and Q. Jiang, "Application of feature engineering for phishing detection," *IEICE Trans. Inf. Syst.*, vol. E99D, no. 4, pp. 1062–1070, 2016, doi: 10.1587/transinf.2015CYP0005.

[14] F. A. Ghaleb, M. Alsaedi, F. Saeed, J. Ahmad, and M. Alasli, "Cyber Threat Intelligence-Based Malicious URL Detection Model Using Ensemble Learning," *Sensors*, vol. 22, no. 9, May 2022, doi: 10.3390/s22093373.

[15] M. Alazab and S. Fellow, "Malicious URL Detection using Deep Learning."

[16] J. Yuan, G. Chen, S. Tian, and X. Pei, "Malicious URL detection based on a parallel neural joint model," *IEEE Access*, vol. 9, pp. 9464–9472, 2021, doi: 10.1109/ACCESS.2021.3049625.

[17] K. Simran, P. Balakrishna, R. Vinayakumar, and K. P. Soman, "Deep Learning Based Frameworks for Handling Imbalance in DGA, Email, and URL Data Analysis," *Commun. Comput. Inf. Sci.*, vol. 1213, pp. 93–104, 2020, doi: 10.1007/978-981-15-9700-8_8.

[18] G. Tan, P. Zhang, Q. Liu, X. Liu, C. Zhu, and L. Guo, "MalFilter: A lightweight real-time malicious URL filtering system in large-scale networks," *Proc. - 16th IEEE Int. Symp. Parallel Distrib. Process. with Appl. 17th IEEE Int. Conf. Ubiquitous Comput. Commun. 8th IEEE Int. Conf. Big Data Cloud Comput. 11t*, pp. 565–571, 2019, doi: 10.1109/BDCloud.2018.00089.

[19] I. Qabajeh, F. Thabtah, and F. Chiclana, "A recent review of conventional vs. automated cybersecurity anti-phishing techniques," *Computer Science Review*, vol. 29. Elsevier Ireland Ltd, pp. 44–55, Aug. 01, 2018. doi:

10.1016/j.cosrev.2018.05.003.

[20] R. S. Rao and S. T. Ali, "PhishShield: A Desktop Application to Detect Phishing Webpages through Heuristic Approach," in *Procedia Computer Science*, 2015, vol. 54, pp. 147–156. doi: 10.1016/j.procs.2015.06.017.

[21] D. Ranganayakulu and C. C., "Detecting Malicious URLs in E-mail – An Implementation," *AASRI Procedia*, vol. 4, pp. 125–131, 2013, doi: 10.1016/j.aasri.2013.10.020.

[22] R. Chiramdasu, G. Srivastava, S. Bhattacharya, P. K. Reddy, and T. Reddy Gadekallu, "Malicious url detection using logistic regression," Aug. 2021. doi: 10.1109/COINS51742.2021.9524269.

[23] R. Bharadwaj, A. Bhatia, L. D. Chhibbar, K. Tiwari, and A. Agrawal, "Is this URL Safe: Detection of Malicious URLs Using Global Vector for Word Representation," *Int. Conf. Inf. Netw.*, vol. 2022-Janua, pp. 486–491, 2022, doi: 10.1109/ICOIN53446.2022.9687204.

[24] A. Hevapathige and K. Rathnayake, "Super Learner for Malicious URL Detection," *ICARC 2022 - 2nd Int. Conf. Adv. Res. Comput. Towar. a Digit. Empower. Soc.*, pp. 114–119, 2022, doi: 10.1109/ICARC54489.2022.9753802.

[25] C. Dian zi ke ji da xue (Chengdu, Guo jia zi ran ke xue ji jin wei yuan hui (China), Institute of Electrical and Electronics Engineers. Chengdu Section, and Institute of Electrical and Electronics Engineers, *MALICIOUS URL DETECTION USING MULTI-LAYER FILTERING MODEL*.

[26] A. Lakshmanarao, M. R. Babu, and M. M. Bala Krishna, "Malicious URL Detection using NLP, Machine Learning and FLASK," 2021. doi: 10.1109/ICSES52305.2021.9633889.

[27] T. Alsmadi and N. Alqudah, "A Survey on malware detection techniques," *2021 Int. Conf. Inf. Technol. ICIT 2021 - Proc.*, pp. 371–376, 2021, doi: 10.1109/ICIT52682.2021.9491765.

[28] M. Verma and D. Ganguly, "Malicious URL Detection using Machine Learning: A Survey arXiv:1701.07179v3," *Corr*, vol. 1, no. 1, pp. 1281–1284, 2019, [Online]. Available: https://doi.org/10.1145/