Edu-SafeLink: Education Simplified Mobile-Based Malicious URL Detection

Rajeswari Raju[1*], Amirul Hakimi Asmadi[1], Sritharan Sangaran[2], Sharifah Nurulhikmah Syed Yasin[1], Ahmad Ihsan Mohd Yassin[3,4] Omkarra Nair[5], Trishul Nair[5], Swasztika Nair[6]

[1]Faculty of Computer and Mathematical Sciences, University Technology MARA, Cawangan Terengganu, Kampus Kuala Terengganu 21080 Kuala Terengganu, Terengganu, Malaysia

[2]PETRONAS Chemical Olefins Sdn Bhd, PCOGD(M), Malaysia

[3]Microwave Research Institute, University Technology Mara, Malaysia

[4]College of Engineering, University Technology Mara (UiTM), Malaysia

[5] SMK Sultan Omar, DESSO, Dungun, Terengganu, Malaysia

[6]SJKC Kwang Hwa, Dungun, Terengganu, Malaysia

[*]Corresponding Author

DOI: https://dx.doi.org/10.47772/IJRISS.2025.903SEDU0499

Received: 20 August 2025; Accepted: 25 August 2025; Published: 24 September 2025

ABSTRACT

In today's education world, the growing prevalence of phishing attacks, especially those targeting mobile users via deceptive URLs, poses a significant threat to cybersecurity and individual privacy. Existing URL detection methods are largely manual, requiring users to copy and paste suspicious links into web-based checkers, which is an impractical and time-consuming solution during urgent situations. This research introduces Edu-SafeLink, an educational mobile-based application designed to simplify and automate the detection of malicious URLs using lexical text mining and the Random Forest algorithm. Targeting Malaysian educational users who are frequently targeted by phishing. Edu-SafeLink enables real-time URL scanning when a link is clicked, whether from messaging apps or QR codes, by intercepting and analysing it in the background. The app provides instant feedback via push notifications and allows safe redirection to the user's preferred browser, thus minimising user effort and reducing vulnerability. Built using Flutter and trained on multiple educational public malicious URL datasets during the implementation phase, the system extracts lexical features such as URL length, special character frequency, and entropy to predict URL safety. The final model achieved an accuracy of 93.47% across multiple split ratios, with the best F1-score obtained using an 80:20 split during the testing phase. By integrating machine learning with seamless mobile UX and push-based alerts, this study presents an efficient and user-friendly defence against educational phishing. The study contributes to building safer educational digital communities, aligning with SDG 11, the Sustainable Development Goal on Sustainable Cities and Communities, which aims to make cities and human settlements inclusive, safe, resilient, sustainable, and offer future opportunities for expansion through deeper URL content analysis or browser extension support.

Keywords—malicious URL detection, lexical text mining, mobile URL detection application

INTRODUCTION

The development of cybercrime has made cybersecurity an urgent need, especially for mobile users. There has been a significant increase in phishing, a fraudulent online attack that aims to steal sensitive data, including usernames and passwords [1]. The extensive use of social media has exacerbated this situation, and attackers exploit users' trust by connecting them to malicious links, resulting in massive data loss [2]. There were even 13,000 complaints of cybercrime alone in 2019, reaching a total of more than 20,000 in 2021, resulting in losses of RM539 million in Malaysia, as well as RM560 million in 2021 [3]. These concerns underscore the need for enhanced mobile security measures that can mitigate the impact of malicious URLs and phishing campaigns on mobile devices.

Considering the free-flowing media of the mobile technology in contemporary life, smartphones have become optimal sources of information and services. This, however, has given cybercriminals a new source of attack, primarily through phishing and the distribution of malicious links. Push notifications are one such mobile technology that can be used to counteract this, enabling applications to receive real-time information and alert their peers. Firdaus et al [4] acknowledge that push notifications also increase communication effectiveness, besides providing users with relevant information given by the application at the right time. The inclusion of push technology within mobile security programs will therefore serve as a proactive model for enhancing awareness and user reaction to any looming cyber-induced challenges.

A combination of machine learning with text mining is a promising approach in identifying and categorising malicious URLs. The proposed methodology utilises the lexical characteristics of URL strings to identify threats, including domain structure, path tokens, and character patterns [5]. In this regard, Edu-SafeLink, an education mobile-based URL detection system, aims to develop an easy-to-maintain and user-friendly solution by utilising Random Forest classification, with no reliance on features beyond lexical features. Moreover, it seeks to review the detection capabilities of Edu-SafeLink by extracting the lexical features only and gauging them with accuracy scores, thus making it a form of lightweight, permissible, and mobile-applied cybersecurity technology that aligns with Sustainable Development Goal (SDG) 11 by promoting safe digital infrastructure within communities.

LITERATURE REVIEW

Machine learning serves as a vital tool which effectively solves multiple digital world problems with a focus on cybersecurity needs. The security threat environment continues to evolve, yet machine learning techniques efficiently detect and counter security risks [6]. Machine learning detection of malicious URLs represents a critical cybersecurity application, as these URLs account for over half of today's cyberattacks and scams [7].

Random Forest

Random Forest algorithm is an ensemble learning algorithm, meaning that several decision trees are combined to increase the accuracy of the prediction and prevent overfitting problems. This is made possible by creating random samples of the given input data set to build several sets of training data where different decision trees are trained across randomly selected feature subsets [8]. The result of the classification is based on the majority vote of all trees, thus making the model less reliant on the prediction of the weak learners. The algorithm randomises both data samples and features, thereby making your model more diverse and resulting in better generalization to unseen data. Random forest can learn complex patterns found in the data, making it applicable to the area of cybersecurity, where the data may have extremely high dimensionality and be highly variable.

The applicability of Random Forest in malicious URL detection tasks is supported by the fact that it works with large numbers of input variables and captures non-linear linkages without being prone to overfitting. Several critical lexical features, including the length of an address URL, the presence of memorable characters and token design, are effectively recognised by the algorithm, contributing to the identification of phishing and malicious actions [9]. It also provides meaningful outputs as it shows the score of feature importance, which helps a researcher appreciate which features of a URL are most influential in making decisions on its classification [8]. This clarity adds value to both the credibility and the practical application of the model in real security settings. When used in conjunction with lexical feature analysis, the Random Forest algorithm enables the design of a lightweight and real-time system, achieving precise levels of malicious URL detection and can be used to enhance mobile security. Table I presents three classifiers with the highest accuracy results achieved by previous researchers, which informed the decision to use Random Forest for this research.

Table I. Classifier Comparison from Previous Research

Author	Classifier	Accuracy (%)
[6]	Random Forest	98.25
	Logistic Regression	97.62
	Decision Tree	97.54
[7]	Random Forest	92
	Gradient Boost	90
	AdaBoost	90
[10	Random Forest	99
	SVC	99
	k-NN (N=5)	98
[11]	Random Forest	97.35
	Bagging	97.35
	SVM	97.32

Similar Work

Researchers can develop efficient schemes for malicious URL detection by utilising lexical features extracted from the compression of URL strings. The advantage of using this method is that malicious links tend to have distinct lexical properties that are vastly different from those of benign links [7]. Lexical analysis will process faster and more securely since no crawling of URLs is needed, nor do any security blocklists need to be checked. This is the justice which makes it well-suited to lower-weight detection systems.

To achieve better detection, authors have used machine learning (ML) and Natural Language Processing (NLP) as well as integrating both lexical features and machine learning (ML) and Natural Language Processing (NLP) features. Waheed et al. [6] utilised 6 ML classifiers and 3 NLP techniques, achieving an excellent 98.2% accuracy with the Random Forest classifier. Similarly, Joshi et al. [7] illustrated an ensemble classification technique with an accuracy of 92% and an AUC score of 0.98 across various datasets. They showed that it has a powerful classification ability to ascertain the maliciousness or otherwise of URLs.

Other studies further confirm the reliability of the ensemble approach. Decision Trees and Random Forest models achieved an accuracy of 91% in all cases. The highest difference was observed in one model, AdaBoost, while Gaussian Mixture was not very useful, reaching a maximum level of accuracy of 82%. In accordance with Raja et al. [10], a Random Forest model with 100 trees achieved the optimal scores for all metrics, including accuracy, precision, recall, and F1-score, while also providing a relatively short execution time. The robustness of Random Forest was once again demonstrated by Wang [11], as it achieved high performance on both datasets used, with accuracy, precision, recall, and AUC being equal, which reflects its generality independent of the data.

Although the hybrid methods of multiple feature extraction used by previous researchers can achieve high performance, the reliance on feature integration increases the complexity of the models. It slows run times, which defeats the purpose of mobile settings. In comparison, using only lexical features may slightly reduce classification accuracy in exchange for significant increases in speed, efficiency, and practicality, which aligns with the lightweight nature of mobile-based malicious URL detection systems.

METHODOLOGY

For this study, the Adapted Waterfall Methodology has been chosen for implementation, as illustrated in Fig. 1. Modifications have been made to the traditional waterfall software development lifecycle to make it more manageable, and this model is referred to as the Adapted Waterfall Methodology. With this approach, feedback and modifications can be made within reasonable limits as one progress through the various stages. According to Maryadi et al. [12], there are conventional classic stages, with limited iterations in between those stages. Still, the purpose is to fine-tune the output before moving on. This flexibility enhances responsiveness and accuracy to changing requirements, especially where user input is critical [13].

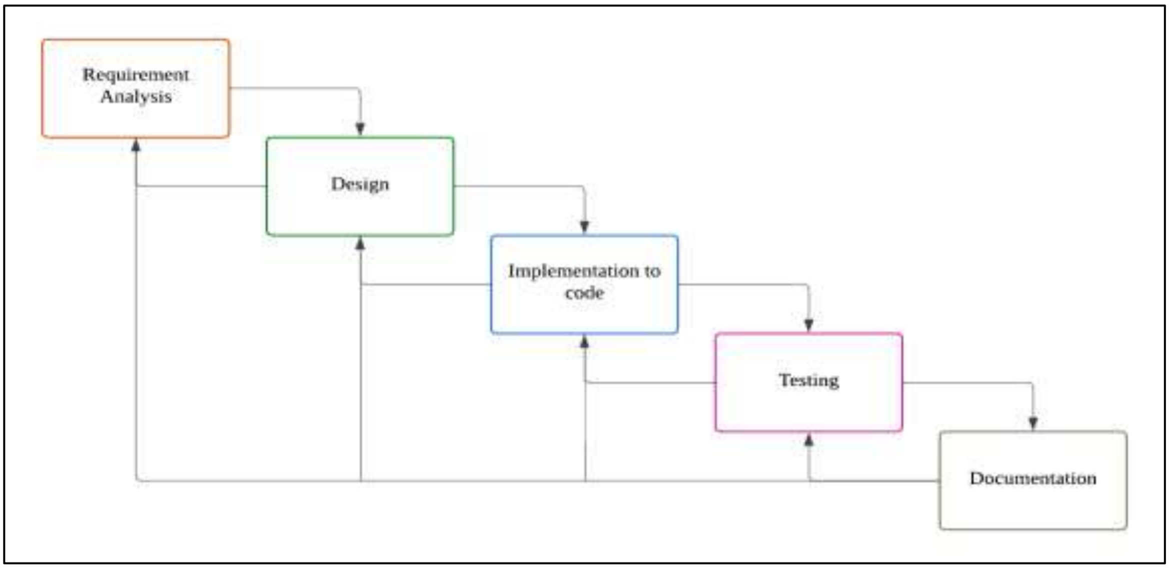


Fig. 1. Adapted Waterfall Methodology

Requirement Analysis Phase

Primarily, the first phase in the adapted waterfall methodology is requirements and analysis. In this phase, user needs and system objectives are understood. Requirements are collected and analysed through document reviews. Documentation at this stage is essential to align user expectations with project deliverables [14].

Design Phase

Next, in the design phase, the requirements are translated into technical models. Unified Modelling Language (UML) diagrams, wireframes, and system architectures are usually employed to describe system components and interactions. This part of the process ensures that the system designed will comply with the specified functional and non-functional characteristics, setting the stage for implementation [15]. Fig. 2 shows the App architecture intended during the design phase.

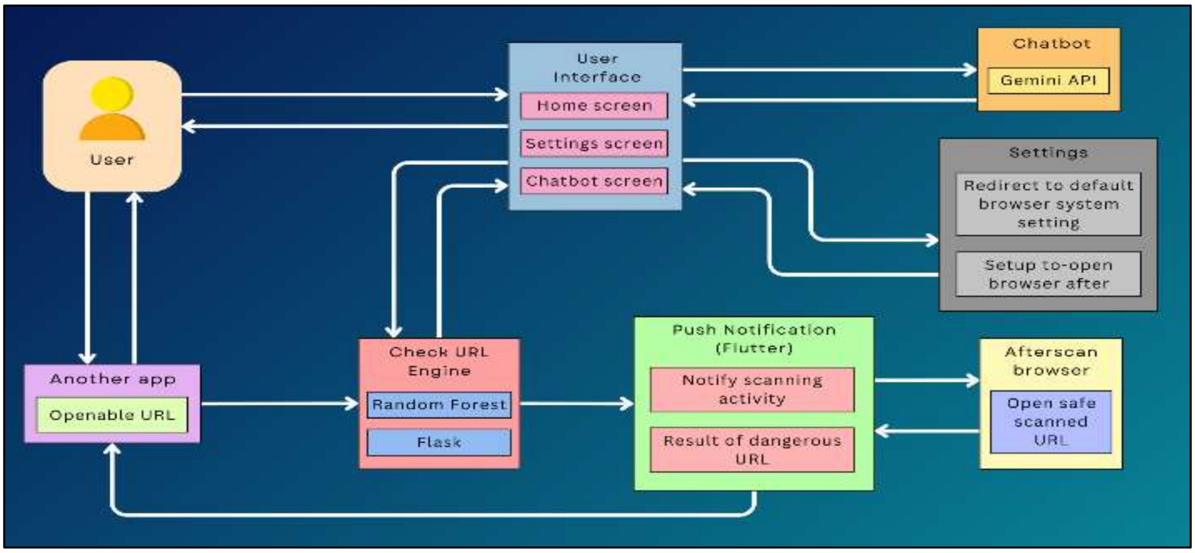


Fig. 2. Edu-SafeLink app architecture

The Edu-SafeLink application employs a unique architectural approach with an optional invisible background scan system. When a user in a different application clicks a link and taps Edu-SafeLink, either by hand or by setting the application as the default browser, it can open with no opacity and a hidden interface. In the background, the intercepted URL is forwarded to the Check URL Engine for verification. With this, the Flutter push notification system alerts the user of the scan being performed and notifies them of the outcome. If the URL is secure, the application automatically opens the preferred browser without any visual break. In case the URL is malicious, a warning sign appears.

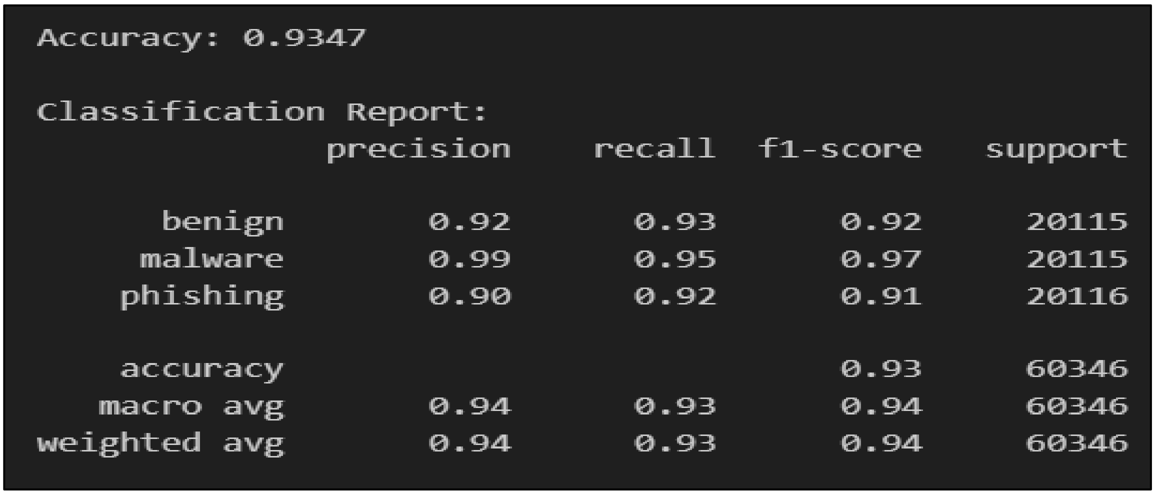
The Edu-SafeLink settings module allows users to access system settings, where they can choose to set Edu-SafeLink as the default browser when configuring their system. There will also be an option to choose which browser to use for verified safe URLs. The AfterScan Browser module also provides additional navigation enhancements by opening safe links in your preferred browser. Its architecture focuses primarily on minimising user friction, being hyper-vigilant against threats, and collaborating efficiently with other Android system components. Edu-SafeLink is well-suited for mobile security applications, as it provides real-time threat analysis without compromising the user experience. This is achieved through its push notification mechanism, cloud-based analysis offloading to Flask, and user transparency.

Implementation To Code Phase

The end of the design phase marks the start of the implementation phase. This is where the design constructed in the system design phase is translated into a software application. Developers utilize the design documents through programming tools and frameworks to build particular modules. This stage may also involve unit testing of software modules to verify correctness before they are integrated with other modules [14]. The split ratios used are 70:30, 80:20, and 90:10 to observe the performance under different proportions of training data. These variations enable the assessment of whether a larger training set significantly improves the classifier's learning or whether diminishing returns occur beyond a certain threshold.

Testing Phase and Documentation Phase

Lastly, the phase concludes with the testing phase and documentation. Testing is conducted to verify if the system meets requirements and operates as intended. To include unit, system, and integration tests so that defects are resolved. Performance testing can be implemented to evaluate the algorithm's accuracy, as shown in Fig. 3. Testing ensures validation, practicality, and efficiency, which are necessary for the success of this system [12]. The report proposal will document the research, including an introduction, literature review, and methodology.



```
Accuracy: 0.9347

Classification Report:
              precision    recall  f1-score   support

      benign       0.92      0.93      0.92     20115
     malware       0.99      0.95      0.97     20115
    phishing       0.90      0.92      0.91     20116

    accuracy                           0.93     60346
   macro avg       0.94      0.93      0.94     60346
weighted avg       0.94      0.93      0.94     60346
```

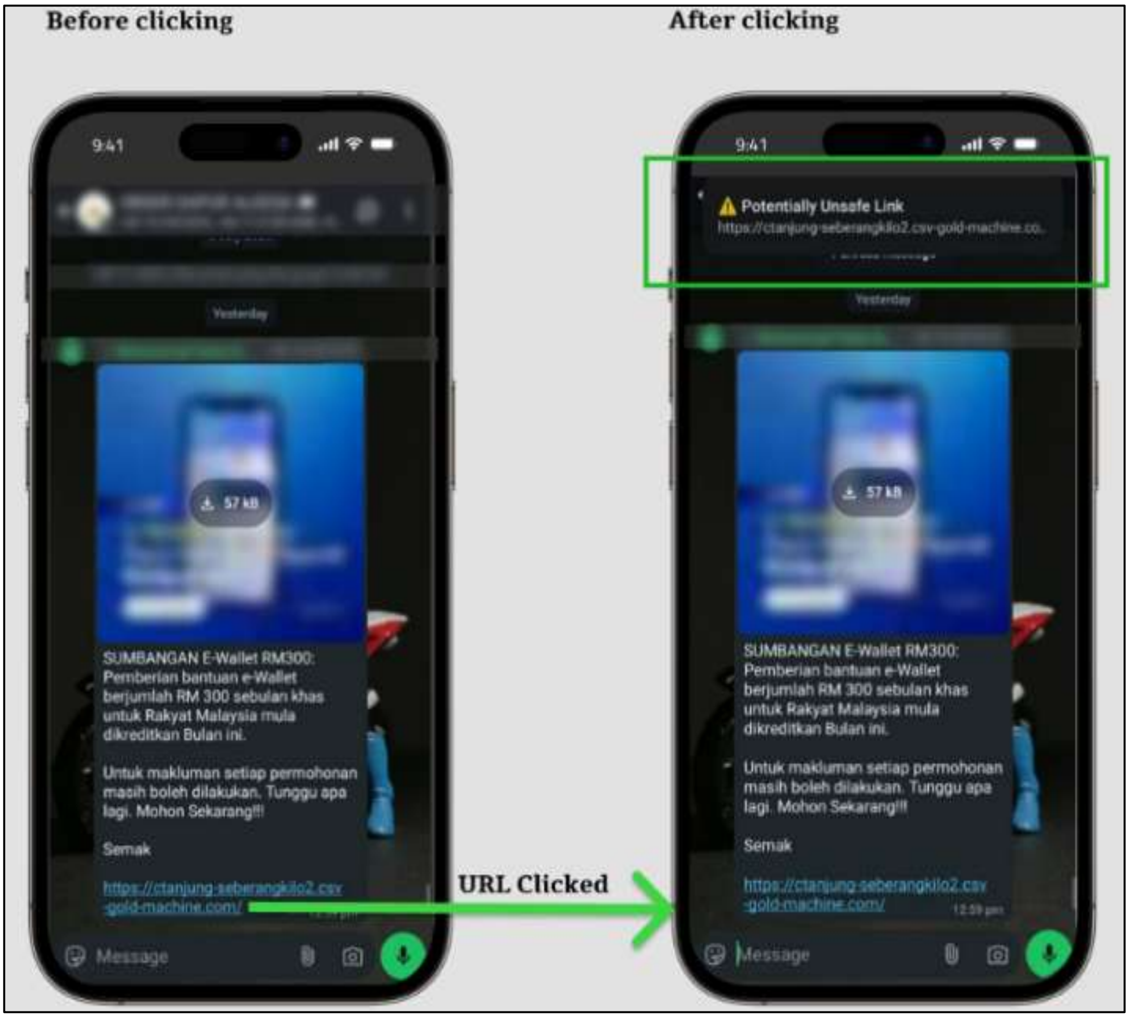
Fig. 3. Performance results of Random Forest using lexical text mining.

RESULTS AND DISCUSSION

The results of the research align with the title, as a simplified prototype version of a malicious URL detection application was developed. Despite certain Android security restrictions, a workaround was implemented to obtain the simplified version. Furthermore, the results of the three split ratios were discussed, with one ratio chosen due to its advantage.

Invisible UI

One of the constraints of the security model in Android is that web URLs cannot be entirely parsed in the background by the app registered to process them, except when it launches them to display a visible interface. Edu-SafeLink performs in this regard with a design strategy that launches a very minimal intervention activity via a transparent theme, making an invisible interface to be effectively started, as in Fig. 4. The user never receives the UI or the sense of an activity happening, as when an activity is technically launched, the illusion of a silent background task is preserved. The activity is supplied with a URL, performs redirection checks, and even conducts safety analysis using an API call, all without requiring overlays or UI components in the Dart code. Then, the user is informed of the outcome using notifications. This will ensure adherence to Android's intent-handling policies without compromising the user experience; for example, the user will not be redirected out of apps while using WhatsApp. The design strikes an ideal compromise between usability, privacy, and platform restrictions by incorporating real-time, non-intrusive multi-URL detection of malicious URLs as a feature.



⚠ Potentially Unsafe Link
https://ctanjung-seberangkilo2.csv-gold-machine.co...

SUMBANGAN E-Wallet RM300:
Pemberian bantuan e-Wallet
berjumlah RM 300 sebulan khas
untuk Rakyat Malaysia mula
dikreditkan Bulan ini.

Untuk makluman setiap permohonan
masih boleh dilakukan. Tunggu apa
lagi. Mohon Sekarang!!!

Semak

https://ctanjung-seberangkilo2.csv-gold-machine.com/

SUMBANGAN E-Wallet RM300:
Pemberian bantuan e-Wallet
berjumlah RM 300 sebulan khas
untuk Rakyat Malaysia mula
dikreditkan Bulan ini.

Untuk makluman setiap permohonan
masih boleh dilakukan. Tunggu apa
lagi. Mohon Sekarang!!!

Semak

https://ctanjung-seberangkilo2.csv-gold-machine.com/

Fig. 4. Screenshots of before and after clicking the URL from another app

Split Ratios Results

The evaluation of various models with three varying ratios between train and test sets (70:30, 80:20, and 90:10) yields essentially similar levels of accuracy, which are 93.32%, 93.47%, and 93.46%, respectively. These slight variations in accuracy suggest that the ratio of the split does not significantly impact the overall gradient of the model in this group of malicious URL assignments. The higher level of examination of the precision, recall, F1-

scores, and confusion matrices, however, allows for a better understanding of the modus operandi of this model, which can be linked to the word clouds created based on each of the classes.

Table II. Accuracy And F1-Score Comparison of Each Split

Split	Accuracy	Malware F1	Phishing F1	Benign F1
70:30	0.9332	0.97	0.91	0.92
80:20	0.9347	0.97	0.91	0.92
90:10	0.9346	0.97	0.91	0.92

Based on Table 1, an 80:20 split ratio was chosen due to its high accuracy compared to the other divided ratios. In addition, the values obtained from malware can be referred to as high-scoring status once again because of the unique concentration of the vocabulary noticeable in the malware word cloud as shown in Fig. 5. While demonstrating some unique tokens such as verify and account, the phishing WordCloud still has many common generic words with benign URLs as shown in Fig. 6 and Fig. 7, so it can be expected that its precision and recall will be slightly lower than the similar previous research, owing to the increased potential of being mislabeled. Additionally, the results of the trained model are marginally lower than those achieved by previous similar work. This is due to the feature extraction, where all cited previous works employed multiple feature extractions. In contrast, this research focuses solely on one type of feature extraction: lexical features, aiming to improve speed and simplicity.

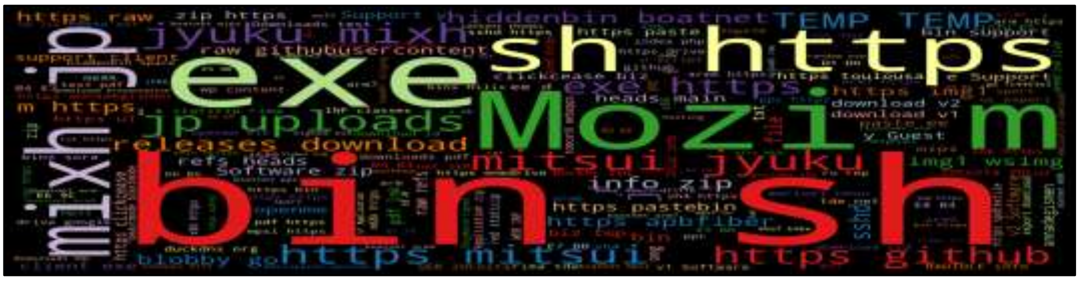


Fig. 5. Malware class word clouds

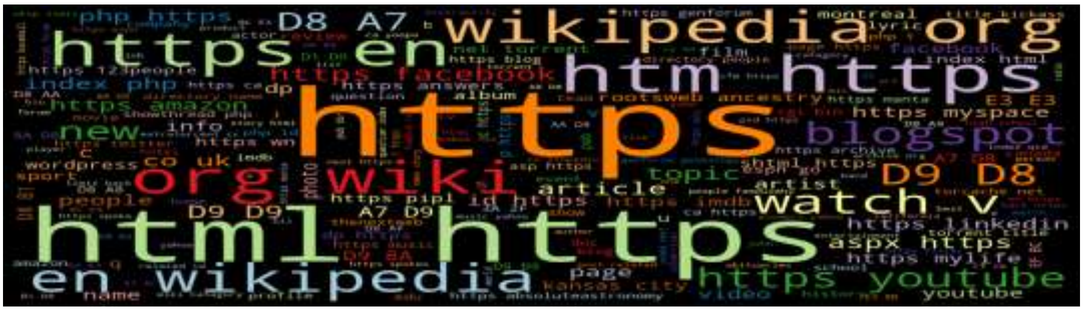


Fig. 6. Benign class word clouds

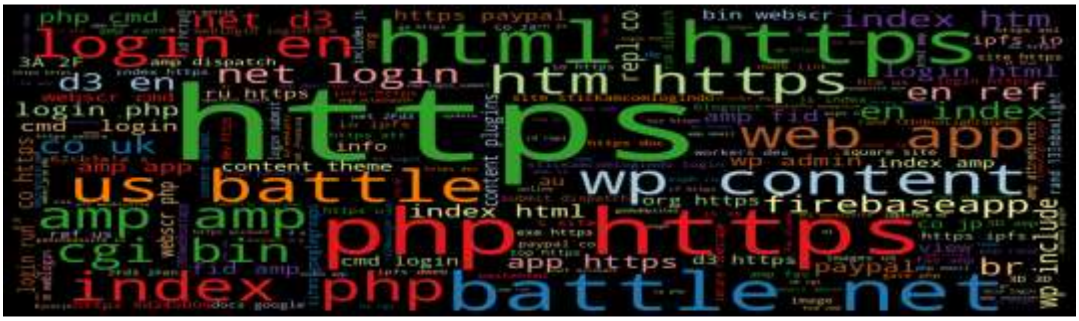


Fig. 7. Phishing class word clouds

CONCLUSION

This study focuses on the increasing risk of phishing attacks that utilize misleading IP addresses to target mobile users and steal sensitive data. Traditional methods of checking URLs are typically manual and cumbersome, especially in situations where quick decisions need to be made. To avoid these drawbacks, Edu-SafeLink was created as a mobile application that can automatically identify malicious URLs using machine learning and lexical feature analyses. The study focused on identifying the weaknesses of modern detection methods, investigating the possibility of combining mobile technologies with the Random Forest algorithm, creating an easily usable mobile application, and analysing its effectiveness in terms of detections. The development process was based on a modified waterfall model, which included data preprocessing, lexical feature extraction, model training, and the development of a mobile application. This novelty was trained on various malicious URL datasets using features such as the length of a URL, the frequency of its characters, the entropy of URL characters, and suspicious words in the URLs, among others.

The resulting application, developed using Flutter, features URL interception in the background, push notifications, and a minimalist user experience. This model was trained and tested using an 80:20 data split and performed well in real-time on mobile devices to identify threats in URLs. The study supports SDG 11 by strengthening digital safety infrastructure in urban environments, thereby contributing to the creation of more inclusive, safe, and resilient communities.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to colleagues, teachers, and mentors at UiTM, Petronas Chemical Sdn Bhd, SMK Sultan Omar, and SJKC Kwang Hwa for providing the necessary resources and support throughout this study.

REFERENCES

1. K. S. Gupta and K. Jayant, "A review study on phishing attack techniques for protecting against attacks," Globus: An International Journal of Management & IT, vol. 10, no. 2, pp. 22, 2019. 10.46360/globus.220191003
2. P. K. V., S. B. H. Shirahatti, S. D. T., S. O. Umair, and S. W. Ahmed, "Phishing - a common cyber menace to combat," International Journal for Research in Applied Science and Engineering Technology, vol. 12, no. 4, pp. 2307–2310, 2024. 10.22214/ijraset.2024.60283
3. S. S. B. Muharram, M. Z. Suhaimi, and M. Marcus, "Cybercrimes in Malaysia," Journal of Education and Social Sciences, vol. 22, no. 1, pp. 34–38, 2022.
4. D. Firdaus, B. Priambodo, and Y. Jumaryadi, "Implementation of push notification for business incubator," International Journal of Online and Biomedical Engineering (iJOE), vol. 15, no. 14, pp. 42–53, 2019. 10.3991/ijoe.v15i14.11357
5. M. Mehndiratta, N. Jain, A. Malhotra, I. Gupta, and R. Narula, "Malicious URL: Analysis and Detection using Machine Learning," in 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 2023, pp. 1461–1465.
6. M. A. Waheed, B. Gadgay, S. DC, V. P., and Q. U. Ain, "A Machine Learning approach for Detecting Malicious URL using different algorithms and NLP techniques," in 2022 IEEE North Karnataka Subsection Flagship International Conference (NKCon), Vijaypur, India, 2022, pp. 1–5.
7. A. Joshi, L. Lloyd, P. Westin, and S. Seethapathy, "Using lexical features for malicious URL detection -- a machine learning approach," arXiv, 2019. 10.48550/arxiv.1910.06277
8. J. Chen, Z. Hu, and Z. Qian, "Research on malicious URL detection based on random forest," in 2022 IEEE 14th International Conference on Computer Research and Development, pp. 30–36, 2022. 10.1109/iccrd54409.2022.9730451
9. B. Praba, K. R. Duddukunta, V. S. Bezawada, and S. V. Addanki, "Enhancing Web security through Machine Learning: A Random Forest approach to Malicious URL Detection," in 2024 4th Asian Conference on Innovation in Technology (ASIANCON), 2024. 10.1109/asiancon62057.2024.10837992

10. A. S. Raja, R. Vinodini, and A. Kavitha, "Lexical features-based malicious URL detection using machine learning techniques," Materials Today: Proceedings, vol. 47, pp. 163–166, 2021. 10.1016/j.matpr.2021.04.041
11. Y. Wang, "Malicious URL detection: An evaluation of feature extraction and machine learning algorithm," Highlights in Science, Engineering and Technology, vol. 23, pp. 117–123, 2022. 10.54097/hset.v23i.3209
12. T. H. T. Maryadi, R. Priyambudi, R. Badarudin, and M. K. B. Martono, "The design and performance of QR-based recognition software for distribution station components," Jurnal Pendidikan Teknologi Dan Kejuruan, vol. 30, no. 1, pp. 62–77, 2024. 10.21831/jptk.v30i1.71438
13. M. T. Amron, N. H. M. Noh, G. J. Tan, and M. A. Ramlan, "Application of the waterfall method in the design of UITM Integrated Staff Information System (ISIS)," International Journal of Advanced Research in Education and Society, 2022. 10.55057/ijares.2022.4.3.3
14. B. A. Nugroho and A. Izzah, "Implementation of Google's technology in Android Mobile App "Kediri City FaSUM and FASOS Information System"," Kinetik Game Technology Information System Computer Network Computing Electronics and Control, pp. 45–56, 2017. 10.22219/kinetik.v3i1.543
15. M. Mokhtar, N. H. A. Hamid, and N. Nazri, "SEATROBS: Development of Sealife Travel & Tour Online Booking System Using Usability Theory," ResearchGate, 202