



Research Proposal Title

Machine Learning-Based Detection of Phishing URLs Using Lightweight Classification Models

Student Name: Md. Ariful Islam

Student ID: 2431309

Programme: MSc Computing Portfolio

Module: Research Methods and Professional Development

Module Code: ACFA7004

University of Wales Trinity Saint David

December 19, 2025

Contents

1	Abstract	2
2	Introduction	3
2.1	Context and Background	3
2.2	Research Problem	3
2.3	Research Aim and Objectives	4
2.4	Significance of the Research	5
3	Literature Review	5
3.1	Background of Phishing URL DetectionStudies	5
3.2	Phishing Detection with MachineLearning Models	6
3.3	Lightweight Models and Feature Engineering	7
3.4	Identified Research Gaps	7
3.5	Positioning of the Present Study	8
4	Research Design and Methodology	8
4.1	Research Design	8
4.2	Proposed Detection Algorithm	10
4.3	Research Methods	10
4.4	System Architecture	11
4.5	Data Analysis and Evaluation Techniques	12
4.6	Research Materials	13
5	Expected Outcomes	14
5.1	Anticipated Results	14
5.2	Implications of the Study	15
5.3	Limitations and Challenges	15
6	Timeline	16
7	References	17

1 Abstract

Phishing attacks continue to be a huge problem in cybersecurity with the use of malicious URLs as bait and deception tactics for users to reveal sensitive content. While machine learning-based approaches have performed well on the task of phishing URL detection, a majority of these methods require heavy models with large computational cost. This makes them unsuitable for such real-time applications on resource-constrained platforms (like web browser, mobile or IoT devices). In this study, we attempt to benchmark lightweight machine learning models with regards to phishing URL detection, emphasizing trading off between classification accuracy and computation time.

The research takes a Design Science Research, experimental oriented method. We will rely on public PhishTank, OpenPhish, and Kaggle datasets to build labeled sets of phishing URLs and real URLs. Light features of words(pairs) and selected host-based features will be computed to reduce runtime. We will train and cross-validate the classifiers: Logistic Regression, Naïve Bayes, Decision Trees, and tuned variant of Gradient Boosting. We will evaluate performance using accuracy, precision, recall, F1-score, inference time and model size.

The anticipated result is in the form of a lightweight phishing URL detection model for real-time application and to support deployable anti-phishing solutions.

2 Introduction

2.1 Context and Background

Phishing has grown to be one of the largest and most successful cyber threats on the global scale, with adversaries resorting more to obscuring URLs in order to compromise user credentials, financial information along with sensitive personal data [1]. Recent threat intelligence this has revealed that there is an annual increase in the number of phishing attacks undertaken, indicating increased scale and complexity [2]. Given that the majority of phishing exploits are based on forged or disguised URLs, URL-based detection is key in blocking users from accessing malware-infected websites before any content was fetched [3].

Machine learning (ML) has also been proved to be an effective method for phishing detection, since ML models can automatically learn distinguishable patterns that exist in lexical, host's information and structures of URLs [4]. Conventional ML models like Random Forest, Logistic Regression, Gradient Boosting have achieved high accuracy but are mostly computational-intensive and not so conducive for real-time detection requirement applications - on a browser, email server or any handheld devices [5].

To mitigate these efficiency challenges, lightweight ML classification models are attracting more and more research interest for reducing computational cost while still achieving high detection performance [5]. The emergence of a large amount of high-quality datasets — PhishTank [14], OpenPhish [15], and Kaggle phishing URL repositories [16] has also contributed to the rapid advancement in these models by facilitating ample training and evaluation [6]. Lightweight design is especially significant for applications such as embedded systems, IoT devices, browser add-ons, and edge-based security functions where resources are limited and threats must be detected fast and efficiently [5].

2.2 Research Problem

Phishing attacks are getting more sophisticated, and it has enabled themalicious URLs to bypass the rules based on blacklists / rule-patterns or signatures that have been pre-defined [1]. Baseline detection systems, however, are incapable of addressing such new or transient malicious URLs and fail to recognize many novel or obfuscated URLs, potentially exposing users to new propagation techniques [2].

Learning lexical and structural properties of phishing URLs, machine learning models have shown high detection performance [3]. Nevertheless, most high-accuracy ML models are resource-hungry to the point that heavy processing power and invite latencies. This causes challenges in instant execution environments like browser extensions, email gateways, mobile and IoT platforms which require low latency and efficiency to be used effectively [4]. Lightweight ML models have however been introduced to solve the problem at stake, their performances are quite disappointing and there is no base approach for selecting pertinent lightweight URL features [5].

Furthermore, most of the existing widespread phishing datasets suffer from repetitions, expired links and/or unbalanced classes that also affect performance and generalisation [7]. Several analyses are not also tested in real-world or real-time environments, which is not prone to be employed for practical cybersecurity proceedings. In addition, existing approaches may have difficulty generalizing to novel phishing behaviors and patterns that are issued every hour, minute or second (e.g.,) 10, motivate the searching for a more robust, efficient and generalizable lightweight detection mechanism [6].

2.3 Research Aim and Objectives

Research Aim:

The goal of this study is to propose and test small machine learning classifiers to detect phishing URLs without sacrificing precision for model size, so that they can be used in-memory instantly on resource-poor devices such as browsers, mobile devices, or IoT platforms.

Research Objectives:

For the pursued aim, the study will follow these objectives:

Analyze available methods for phishing URL detection and point out the limitations related to the performance of techniques, efficiency and real-time use [4, 7].

To create a well-annotated dataset containing both phishing and legitimate URLs from credible public sources (e.g., PhishTank, OpenPhish).

To remove and design lightweight lexical, as well as host-based features to reduce computational load without affecting the classification results [5].

To develop and evaluate several simple but effective ML models such as LR, Decision Trees (DT), Random Forests (RFs), Naïve Bayes (NB) or Gradient Boosting Machine in

their different implementations to identify the best-performing approach.

In order to compare the model’s performance based on accuracy, precision, recall, F1-score and inference time focusing on real-time detection (Nguyen et al., 2013).

To propose an appropriate lightweight model for real-world usage in low-resource or real-time security scenarios.

2.4 Significance of the Research

Violations against URL standards Evasion of detecting Phishing remains major . . . threat- Large exposure risk pattern deceiving emerging Increasingly complex URL Phishing deceptive Filter evasion attacks continue to be a serious computer security threat [1]. Lightweight features targeting for the protection against phishing URLs are important as they allow fast, real-time detection in low computation devices e.g. mobiles devices browsers IoT platforms [2]. The state of the art accurate ML models already demand high computational power and are not viable to be deployed in these conditions [4].

This work offers contribute to the state of the art by studying also light feature set- sand classified methods with reasonable trade-off between accuracy and computational complexity. By benchmarking the model performance with real-world datasets and paying attention to inference time, this work offers practical deployment considerations for security applications [5]. The results will advance the state of the art in scalable anti-phishing solutions in contemporary distributed systems, edge computing environments, and large-scale online platforms [6]. In the end, it is hoped that this research will lead to increased user protection through more timely and broadly deployable phishing detection on users’ machines and networks.

3 Literature Review

3.1 Background of Phishing URL Detection Studies

Phishing attacks are underpinning one of the most insidious and fast-evolving cybersecurity challenges, often using malicious URLs to deceive users into disclosing sensitive information such as account credentials and financial details [APWG2023]. Phishing attacks are getting more advanced, for example the attackers use tricks such as typosquatting,

homoglyph replacement, URL shortening and domain obfuscation to bypass rule-based detection and blacklist-based methods [8]. As a result, there has been an increasing demand for smart attack detectors that can recognize malicious URLs prior to the user interacting with the dangerous material.

URL phishing detection has been receiving increasing attention due to its efficiency and early-defense nature. URL analysis, unlike content-based methods, does not involve retrieval or rendering of web-content which substantially lowers the computational expenses and systemic vulnerability to malicious scripts [4]. This makes URL-based detection even more applicable to security applications in real-time. Recent works have extracted discriminative features from URLs, such as lexical, structural, and host-based features to have higher classification accuracy with lower complexity.

Lexical features: The URL length, character distribution, suspicious tokens, symbol frequency and the entropy settings are mostly used as obvious feature (s) for their simplicity and effectiveness [9]. Derivative attributes such as subdomain depth, domain age and DNS information can provide further context but may add overheads of latency and external service reliance constraining applicability in time-crucial operations [7]. The trade-offs from above that emphasize the need for lightweight detection strategies that trade performance for efficiency.

3.2 Phishing Detection with Machine Learning Models

Phishing URL detection has been increasingly dominated by machine learning methods because it can automatically learn from a huge amount of example data (i.e., complex patterns). The most common classifiers are simply the Logistic Regression, Naive Bayes, Support Vector Machine and Decision Tree, Random Forests and Gradient Boosting models [3]. Ensemble methods such as Random Forests and Gradient Boosting often achieve a strong detection accuracy by recording non-linear feature interactions and averaging variance [10]. Nevertheless, the long inference time, memory overhead and model complexity make them challenging to deploy in real-time on resource-limited devices [4].

More simple classifiers such as Logistic Regression and Naïve Bayes, provide a faster inference and require less computation, which are interesting for light-weighted detection problems [11]. Decision Trees also offer efficient classification along with a set of interpretable decision rules, which might be advantageous for the security analyst. Al-

though they have benefits, the utility of these models is heavily reliant on feature quality, dataset balance and evaluation strategy that differ significantly amongst studies[8]. Furthermore, models trained on snapshot datasets may not generalise to new phishing tactics that develop in response to network defender interventions and will decay over periods of time [6].

3.3 Lightweight Models and Feature Engineering

There has also been a growing interest on light-weight ML models as an efficient approach to phishing detection in resource-constrained environments, for example browsers, mobile devices and IOT systems [6]. These models tend to value low complexity, fast inference and small memory footprint at the cost of acceptable detection accuracy. The lexical feature based techniques are particularly suitable for lightweight models because they do not involve any network communication, and also the constraints of depending on external data [9].

Feature engineering is an essential factor for detection performance and efficiency improvement. Various feature selection methods such as information gain, correlation-based filtering or chi-square analysis can be employed for reducing redundancy and computational load [3]. Host-based and content-based features may improve detection performance but are neglected in lightweight systems to avoid overhead, privacy and potential risks [11]. As a result, large amount of research concentrate on efficient lexical features to facilitate scalable and real-time phishing detection.

3.4 Identified Research Gaps

While significant advances have been achieved, several gaps in research remain. Although many state-of-the-art phishing detection models are available, most of them still not feasible for deploying on time/real time scenario because of computational limits [4]. Challenges like outdated URLs, duplicated samples and class-imbalanced datasets still hinder model robustness and generalisability [7]. In addition, majority of the research focuses on accuracy as a most important aspect neglecting measures that are important for practical deployment such as inference time, memory usage and model size [6]. The multidisciplinary nature of this subject has also contributed to the missing benchmarking frameworks for lightweight feature sets in an open-source form, which adds to

maintain the relationship among studies [3].

3.5 Positioning of the Present Study

In this work, we attempt to fill these gaps by performing a systematic investigation of lightweight machine learning classifiers enabled by optimised URL feature sets under realistic CPU-based constraints. The paper compares performance and efficiency metrics among several classifiers with the goal of identifying deployable phishing URL detection solutions that balance high accuracy against computational cost. This helps in designing scalable real-time anti-phishing systems applicable to today’s cybersecurity landscape.

4 Research Design and Methodology

4.1 Research Design

The approach in this study is Design Science Research (DSR) method that is commonly used in the computing and cybersecurity areas to help develop and evaluate technological artefacts such as machine learning based detection models [12]. DSR is correct for this project as it aims to solve a real-world problem—i.e., Better efficiency and real-time usability of Phishing URL detection using light weight machine learning models). The design of the detector consists in identifying the problem (appearance-only people detection), defining the requirements, building the detection model and evaluating its performance by systematic experimentation.

The research approach of the study also includes an experimental one, to validate the DSR framework. Experimental approaches are particularly useful for machine learning research since they permit controlled comparison between models, sets of features and evaluation metrics across a range of benchmark datasets [13]. This method likewise makes it possible to compare lightweight models such as Logistic Regression, Decision Trees, and Naïve Bayes against optimized Gradient Boosting ensemble variants in order to establish which models present the best balance between accuracy and computational cost.

In general, the research design is comprised of four stages. First, we gather a corpus of URLs over phishing reports and normal ones from well-known public sources such as

DSR Stage	Main Activities	Results
Problem Identification	Review the literature for identification of phishing URL; point out the shortfall in computational effectiveness and real-time usability [12]	well-defined research problem and objectives.
Requirement Definition	Decide on software requirements, features and metrics	Design requirements for effective phishing detection
Design and Development	Feature extraction (lexical and host-based); model implementation with lightweight classifiers (Logistic Regression, Naïve Bayes, Decision Trees)	Designed low-complexity phishing-detection models
Experimental Evaluation	Training/testing the model on benchmark datasets; performance evaluation in terms of accuracy, F1-score, and inference time [13]	Comparative performance results
Analysis and Validation	Analyse trade-offs between accuracy and computational efficiency; evaluate real-time availability	Validated Lightweight Model for Deployment

Table 1: Summary of the Research Design Framework

PhishTank, OpenPhish or Kaggle. Second, we extract and engineer light-weight lexical and host features to create a compact feature set. Third, a nested cross validation scheme is adopted to train a variety of light-weight machine learning algorithms and performance are benchmarked (accuracy, precision, recall, f1-score), and inference time. Finally, the performance of the above models are analysed to determine which lightweight model is better for deploying in real-time and embedded situations. The proposed approach is organized so that the research aims are addressed in an organized manner, and allows for a systematic assessment of the models produced.

4.2 Proposed Detection Algorithm

Algorithm 1 Lightweight Phishing URL Detection Algorithm

- 1: Gather phishing and nonphishing URL datasets from public sources
 - 2: Preprocess the URLs by eliminating duplicates and invalid entries
 - 3: Derive low-overhead lexical and host-level features from Input URLs
 - 4: Split data to training and test set
 - 5: Fit simpler models (Logistic Regression, Naïve Bayes, Decision Tree)
 - 6: Measure models by metrics of accuracy, precision, recall and F1-score, inference time
 - 7: Choose the most suitable model regarding to accuracy and efficiency ratio
 - 8: Output the lightweight phishing detection model being trained
-

4.3 Research Methods

This work also takes a quantitative methodological stance, because the problem phishing URL detection is based on numerical characteristics, statistics and performance measures. For model evaluations, you will be using quantitative measures for assessing the ML classifier performance, thus it is possible to systematically compare certain classifiers (and their settings) based on various metrics such as accuracy, precision, recall and F1-score as well as other computational-related factors [13]. A mixed experimental approach is used to judge whether the performance of alternative lightweight machine learning models based on optimised URL feature sets.

The experimental protocol consists in perform controlled evaluation over several lightweight classifiers, such LR, DTrees, NBayes and the high-optimised versions Strutz04 boosting. These models are chosen for their efficiency and real-time capability, along with effective classification performance [11]. They train models on a subset of the data and validate them on another part, with some sort of validation/validation process that guarantees stability and reliability.

Supervised training mechanisms are applied because a set of predefined phishing and non-phishing URL labels in the dataset [8]. To reduce the computation cost, lightweight lexical and selected host-based features are first extracted and pre-processed (i.e., normalisation, token cleaning, and feature selection). Evaluation of the models consist both on classification metrics, as well efficiency driven ones: inference time, memory needs, and model size to consider its deploying to real-time and resource limited applications [6].

4.4 System Architecture

The system architecture in this work is an organized machine learning pipeline which has been implemented for lightweight phishing URL detection. The workflow is comprised of five main phases: dataset collection, preprocessing, feature set generation, model training and performance assessment. It is designed so that each stage collaboratively optimizes and incrementally improves the classification performance for both real-time processing and resource-limited scenarios [6].

Phishing vs legitimate URLs **phishingsitesonline dataset** The first stage is to extract phishing and legitimate URLs from popular public repositories (eg: PhishTank, OpenPhish, Kaggle). Labelled data from these sources can be used as input for bourse-ction [8]. The collected dataset is preprocessed by removing duplicate URLs, missing data and noisy entries. Token normalization, cleaning and slashing irrelevant characters are used to standardize URL strings before we perform feature extraction.

At the feature engineering step, we pay attention to create lightweight lexical and host-based features which can be easy to process. Features such as URL length, number of subdomains, symbol frequency, and entropy measures are computed from the URL string directly [9]. Host-based attributes over these sorts of selection including domain age and DNS are included only if the computational cost is not prohibitively high. Feature selection can be used to overcome redundancy and improve optimality of a model [3].

In the model training phase, several lightweight machine learning classifiers are trained with the hand-crafted feature space; they include Logistic Regression [10], Decision Trees (DTs) [11], Naïve Bayes (NB) and optimized variants of Gradient Boosting Trees (GBT). Cross-validation is also applied to determine the stability of each model and reduce over-fitting [13]. Then, we evaluate classification accuracy and efficiency (such as inference time, memory usage, and model size) for the trained models at the performance evaluation stage. The signs show the extent to which models are applicable for real-time identification systems [4]. The least step is to evaluate models and determine the best light-weight classifier for phishing URL detection.

This design offers a unified and scalable workflow for training, analyzing, and benchmarking lightweight phishing classifiers. Each step is clearly related to the purpose of this research, that is, finding a model with high accuracy and low complexity.

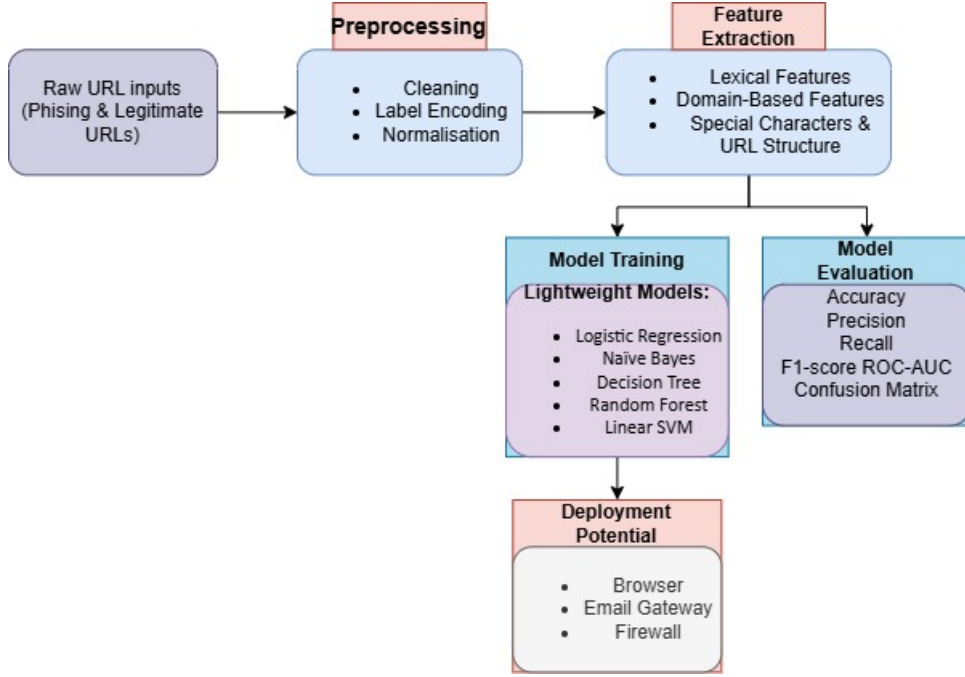


Figure 1: The Proposed System Architecture for Lightweight Phishing URL Detection

4.5 Data Analysis and Evaluation Techniques

In this work we analyze how lightweight machine learning models can be effective for phishing URL detection. The data have been split into the training set, and testing in order to provide an unbiased measure of model performance. Stratified sampling is employed to achieve proportionate representation of phishing and bona fide URLs, reducing the class imbalance usually present in phishing datasets [7]. Cross-validation is also employed for additional validation of model stability and to diminish over-fitting [13].

We measure model performance by accuracy (overall, per-class), precision, recall and F1 score with standard models establishing the benchmarks. These indicators have been extensively employed in the literature of phishing detection to measure a model’s performance on discriminating malicious from benign URLs [12]. In this area precision and recall are very important as high proportion of false positives can disrupt legitimate usage, while high proportion of false negatives to miss those phishing attacks. Balanced performance measurements are offered by the F1-score.

Besides classification performance, the computational computing time is evaluated for real-time application. Efficiency metrics include inference time, model size and memory footprint, on which models run well in resource-limited platforms (e.g., browsers, IoT devices and mobile phones) [6]. Where applicable, statistical testing is also used (paired

t-test or Wilcoxon signed-rank test) to determine the significance of performance differences. This comprehensive evaluation will jointly consider each model’s applicability as a real-world tool.

Metric Type	Metric Description
Classification Metrics	Accuracy, Precision, Recall, F1 Score
Efficiency Metrics	Inference Time (ms), Model Size (KB), Memory Usage (MB)
Robustness	Cross-Validation Scores and Statistical Significance Tests

Table 2: Evaluation Criteria in the Paper

4.6 Research Materials

This paper uses open-source academic and research resources suitable to develop lighter machine learning models for the problem of phishing URL identification. Both phishing and legitimate URLs are from PhishTank, OpenPhish, Kaggle and so on that contain the labelled dataset for extensive activities of phishing detection research such as [8][1] pulling updated samples based upon current attack trends. Preprocessing of the dataset consists in filtering duplicate and inactive URLs, and validation of labels for checking the quality of data.

We develop the classifiers in Python using Scikit-learn library which are light-weight and include: Logistic Regression (LR), Naive Bayes, Decision Trees, and Gradient Boosting variants [14]. Data preprocessing and feature engineering are supported in both Pandas and NumPy, with performance visualisation performed in Matplotlib/Seaborn.

Model training and evaluation are performed on a standard CPU personal computer (equivalent to Intel i5 with 8–16 GB RAM) representing the emphasized real-time operation environment and lightweight requirements of this class of models [6]. Ethics considerations are relatively low, since only publicly available URL datasets are exploited; no personal data is included; and the feature extraction takes place at the metadata and lexical levels. In summary, the used materials make it possible to achieve replicated and economic research reflecting our goal of obtaining lightweight adequate models for phishing URL detection.

Task Purpose	Material
PhishTank Dataset	Furnishes the set of labelled phishing URLs for supervised learning and model building
OpenPhish Dataset	Supplies up-to-date phishing URL samples reflecting current attack patterns.
Kaggle Datasets	Phishing mixed phishing and URLs for training and testing Kaggle Datasets offers
Python Programming Language	Primary environment for data processing, model implementation and model evaluation
Scikit-learn	Provides easy-to-use machine learning classifiers and metrics
Pandas & NumPy	Data preprocessing and cleaning Feature engineering
Matplotlib & Seaborn	Create plots of model performance and evaluation
Personal Computer (CPU-powered)	Performs model training and evaluation in a computationally challenged system

Table 3: Materials and Tasks

5 Expected Outcomes

5.1 Anticipated Results

The key anticipated deliverable of this proposal will be to: Identify and validate lightweight machine learning models that can efficiently detect phishing URLs with high detection quality in real-time. According to the existing literature, classifiers including Logistic Regression, Naïve Bayes and shallow Decision Trees should provide competitive performance with low computational cost[11]. We expect these models to yield accuracy and F1-score closely competitive of more complex ensemble-based methods while obtaining a much lower inference time, memory footprint and model size[6]. Therefore, the chosen lightweight techniques should be especially suitable for resource-limited settings like browser extensions, mobile phones and IoT devices.

Secondly, URL features extracted from lexical attributes are also expected to dominate the detection performance. Lexical features are computationally efficient and helpful in identifying the irregular structural patterns that often emerge in phishing URLs[9]. Although host-based or external features could provide some performance gain, they would incur extra retrieval latency leading to decreasing the practicability of these models for real-time detection[7]. Therefore, we expect that the study will show that an optimised subset of lightweight lexical features strikes a functional middle-ground between accurate

detection and speed.

5.2 Implications of the Study

Practical and Academic Contributions The outcomes of this study have both practical and academic implications. Pragmatically, the established lightweight models can be utilized to develop deploy-able anti-phishing solutions and they are very efficient as there is no need of arduous hardware for their operation. These solutions can be incorporated into live cybersecurity systems, such as web browsers, email filters and mobile security products straight away meaning the fighting back against phishing in everyday computing could soon get a whole lot easier.

This work is anticipated to be a meaningful contribution to the research community in terms of verifying the tradeoffs between model complexity, effectiveness and efficiency for phishing URL detection. By using a sound experimental design to benchmark several lightweight classifiers under equal evaluation conditions, this work could contribute to formulating guidelines and reference baselines for future research. Moreover, feature effectiveness analysis can help shape future studies and promote unified lightweight feature sets for more reproducible and comparable research results in the field of phishing detection.

5.3 Limitations and Challenges

However, a number of limitations are expected notwithstanding the predicted benefits. A major issue is also related to the data sets, such as class imbalance, stale URL and with that fake URLs (no longer available), quick changeability of phishing methods etc. These factors could impact on model generalisation, leading to varying performance across datasets as previous works reported [8]. Moreover, the historical solutions can not work well with novel or highly obfuscated phishing URLs utilizing new (unknown) attack techniques.

Another drawback is that only lightweight, CPU-based evaluation scenarios are considered. Although this is in line with how the deployment of the study is intended, it may limit the comparison to more time-consuming deep learning approaches. Moreover, the dependence on URL-based features only might hinder the detection accuracy in

finer-grain when phishing behavior can be more recognized by content-based or contextual analysis.

In general, despite the limitations of this study, a validated lightweight phishing URL detection model is anticipated to be provided and useful insights on the tradeoffs of accuracy, efficiency and real-time practicality in current network defense systems.

6 Timeline

Phase	Activities	Duration (Weeks)
01	Literature Review	Weeks 1–3
02	Dataset Acquisition	Weeks 3–4
03	Data Preprocessing	Weeks 4–5
04	Feature Engineering	Weeks 4–5
05	Model Development	Weeks 5–6
06	Model Evaluation	Weeks 6–8
07	Results Analysis	Weeks 8–9
08	Report Writing	Weeks 9–10
09	Final Review and Submission	Weeks 11

Table 4: Proposed Project Timeline

7 References

References

- [1] Anti-Phishing Working Group (APWG), “Phishing activity trends report,” <https://apwg.org/reports/>, 2023, accessed: 2025-08-28.
- [2] IBM Security, “Ibm x-force threat intelligence index 2024,” <https://www.ibm.com/reports/xforce-threat-intelligence>, 2024, accessed: 2025-08-28.
- [3] O. K. Sahingoz, E. Buber, O. Demir, and B. Dirir, “Machine learning based phishing detection from urls,” *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019.
- [4] R. S. Rao and A. R. Pais, “Detecting phishing websites using machine learning: An updated survey,” *ICT Express*, vol. 6, no. 3, pp. 170–175, 2020.
- [5] M. Althobaiti and A. Alsubaie, “Lightweight phishing detection model using optimized lexical features,” *Computers & Security*, vol. 120, pp. 1–10, 2022.
- [6] T. Nguyen, Q. Le, and H. Pham, “Lightweight machine learning models for real-time phishing url detection,” *IEEE Access*, vol. 11, pp. 45 012–45 025, 2023.
- [7] R. S. Rao and A. R. Pais, “Machine learning techniques for detecting phishing websites: A survey,” *ICT Express*, vol. 7, no. 3, pp. 258–265, 2021.
- [8] A. Basit, H. Zafar, A. Javed, Z. Jalil, and M. Abbas, “A comprehensive survey of ai-enabled phishing attacks detection techniques,” *Telecommunication Systems*, vol. 78, p. 571–605, 2021.
- [9] R. Verma and A. Das, “What’s in a url: Fast feature extraction and malicious url detection,” *IEEE Security & Privacy*, vol. 15, no. 4, pp. 52–59, 2017.
- [10] P. Patil, N. Dey, and A. Ashour, “Ensemble-based machine learning approach for phishing url detection,” *Journal of Information Security and Applications*, vol. 58, p. 102790, 2021.
- [11] Y. Xiao, Z. Pan, and J. Wang, “Phishing url detection using machine learning: A review,” *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 5, p. e3975, 2020.

- [12] A. Hevner, S. March, J. Park, and S. Ram, “Design science in information systems research,” *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.
- [13] D. C. Montgomery, *Design and Analysis of Experiments*, 9th ed. Wiley, 2017.
- [14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, “Scikit-learn: Machine learning in python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.