Machine learning based approach for detecting malicious URLs in social media platforms

By:

Yara Darawsheh – 202220859

Supervisors:

Prof. Farah Zawaideh

Dr. Motasem Abu-Dawas

Submitted in partial fulfillment of the requirements for the Bachelor's Degree in Cybersecurity, Faculty of Science and Information Technology, Irbid National University

Irbid-Jordan

2026

We hereby declare that the work in this graduation project at Irbid National University is our own except for quotations and summaries which have been duly acknowledged. This work has not been accepted for any degree and is not concurrently submitted for award of other degrees. It is the sole property of Irbid National University and it is protected under the intellectual property right laws and conventions.

We hereby declare that this report is our own work except from properly referenced quotations and contains no plagiarism.

We have read and understood the school's rules on assessment offences, which are available at Irbid National University Handbook.

Students:	**Signature:**	**Date:**

Dedication

In the name of Allah, the Most Gracious, the Most Merciful.
May peace and blessings be upon our Prophet Muhammad, and upon his family and companions

All praise is due to **Allah**, always and forever. We thank Him with praise befitting His majesty and greatness, for His countless blessings, His guidance, and His kindness that accompanied us throughout this journey. Without His grace, we would not have reached this stage, nor would this work have been completed.

We dedicate this humble effort to those who, after Allah, played a vital role in guiding and supporting us. Our sincere appreciation goes to our respected supervisor, **Dr. Farah Zawaideh**, who has been a true example of academic dedication, patience, and sincere guidance. We are deeply grateful for his valuable time, insightful advice, and continuous support, which greatly contributed to completing this work in its best possible form.

We also dedicate this achievement to our beloved **family**, the source of strength, security, and unconditional love. They believed in us before we believed in ourselves, and supported us with their prayers, patience, and encouragement through every challenge. This accomplishment is theirs as much as it is ours, for they are the foundation upon which every success is built.

Our gratitude extends to our **friends**, our companions along the journey, who shared with us moments of effort and joy, and stood by us with sincere words and motivating support. Their presence left an unforgettable mark on our path.

Finally, we dedicate this achievement to **ourselves**, in recognition of every effort exerted, every moment of perseverance, and every step taken despite difficulties. This work is the result of determination and hard work, and a testament that sincere effort is never wasted.

We pray that Allah accepts this work, makes it purely for His sake, benefits others through it, and grants it success and distinction. May it be a blessed step in our academic and professional journey. Indeed, He is the All-Powerful and the Most Capable.

LIST OF CONTENT

LIST OF TABELS

LIST OF FIGURES

LIST OF ABBREVIATION

Concept	Abbreviation
Artificial Intelligence	AI
Application Programming Interface	API
Area under the Curve	AUC
Convolutional Neural Networks	CNNs
Comma-Separated Values	CSV
Deep Learning	DL
Domain Name System	DNS
Decision Tree	DT
False Negative	FN
False Positive	FP
Internet Protocol	IP
Long Short-Term Memory	LSTM
Loopy Belief Propagation	LBP
Logistic Regression	LR
Machine Learning	ML
Random Forest	RF
Receiver Operating Characteristic	ROC
Recurrent Neural Network	RNN

Support Vector Machine	SVM
Solid State Drive	SSD
True Negative	TN
True Positive	TP
Top-Level Domains	TLDs
Uniform Resource Locators	URLs
University of New Brunswick	UNB

ABSTRACT

This project focuses on securing social media platforms by detecting malicious URLs using machine learning techniques. With the rapid increase in online content sharing, attackers commonly spread harmful links disguised as shortened or deceptive URLs. These links can lead to phishing pages; malware downloads, or unauthorized data collection. Traditional cybersecurity methods struggle to keep up with these evolving threats because they rely on fixed rules that do not detect new or hidden attack patterns.

To address this challenge, this study develops an intelligent ML-based detection model that analyzes URL properties—including length, domain structure, special characters, and lexical patterns—to identify malicious behavior. The model is trained on a dataset of labeled URLs and evaluated using multiple ML algorithms such as Random Forest, Logistic Regression, SVM, and deep learning models to determine the most accurate classifier. The aim of this study is to create a lightweight, fast, and effective detection tool that can help social media users avoid malicious content.

Experimental results show that the Random Forest classifier achieves the highest detection accuracy, demonstrating its ability to capture complex URL patterns. This project contributes to enhancing online safety by providing a proactive defense mechanism capable of adapting to new cyber threats.

CHAPTER 1

INTRODUCTION

1.1 Introduction to Social Media Platforms and Their Expansion

Social media platforms have become an essential component of modern digital life, profoundly transforming the way individuals communicate, interact, and consume information. Millions of users rely on platforms such as Facebook, Instagram, X (Twitter), TikTok, and LinkedIn on a daily basis to exchange messages, share multimedia content, follow news, conduct business activities, and participate in public discussions. These platforms are no longer limited to social interaction; instead, they play a critical role in education, marketing, politics, and digital economies.

According to recent statistics, Facebook alone has over 3 billion monthly active users, Instagram around 2 billion, TikTok more than 1 billion, and LinkedIn over 900 million users worldwide. These figures highlight the massive reach of social media platforms and their role in shaping global communication patterns. Moreover, during the COVID-19 pandemic, social media usage surged as people relied on these platforms for remote learning, online business, and staying connected with family and friends.

Over the past decade, social media platforms have experienced unprecedented growth driven by several factors, including the widespread availability of high-speed internet, the proliferation of smartphones, and the rise of cloud-based technologies. This rapid expansion has enabled real-time global connectivity, allowing information to spread instantly across geographical boundaries. As a result, social media has become one of the most influential tools shaping public opinion and digital behavior. Furthermore, governments and businesses increasingly use social media to engage with citizens and customers, demonstrating its importance beyond personal communication.

In addition, social media platforms have evolved into open ecosystems that support content creation, targeted advertising, influencer marketing, and data-driven personalization. Advanced algorithms analyze user behavior to recommend content, advertisements, and connections, leading to massive and continuous data flow generated every second. While this data-driven environment enhances user engagement and platform profitability, it also increases the system's complexity and exposure to security threats. Recent studies have shown that the volume of data generated by social media platforms is growing exponentially, with billions of posts, comments, and messages created daily, presenting both opportunities and challenges for cybersecurity.

However, the rapid expansion of social media has introduced significant security challenges. The openness of these platforms, combined with user trust and high interaction rates, makes them attractive targets for cybercriminals. Attackers exploit both technical vulnerabilities and human factors—such as curiosity, trust, and lack of security awareness—to conduct cyberattacks, spread

malicious content, and manipulate users. Consequently, securing social media environments has become a critical concern for individuals, organizations, and platform providers alike. Addressing these challenges requires a combination of advanced technological solutions, user education, and continuous monitoring to ensure a safe and reliable digital environment.

1.2 Cybersecurity Risks and Malicious URLs

Cyberattacks represent one of the most severe and rapidly growing threats facing social media platforms today, with malicious URLs being among the most common and dangerous attack vectors. A malicious URL is a web link designed to deceive users into visiting harmful websites that may host phishing pages, malware downloads, or exploit kits. These URLs are often embedded in posts, comments, advertisements, or private messages, making them difficult to detect through manual inspection.

Attackers employ various deception techniques to increase the success rate of malicious URLs. These techniques include URL shortening services, the use of misleading keywords (such as “free,” “urgent,” or “verified”), spoofed domain names that closely resemble legitimate websites, and multiple redirection layers that hide the final destination of the link. On social media platforms, such URLs spread rapidly due to resharing, liking, and algorithmic promotion.

The consequences of clicking malicious URLs can be severe. Users may unknowingly provide login credentials on fake authentication pages, leading to account takeover and identity theft. In other cases, malicious URLs initiate silent malware downloads, allowing attackers to gain unauthorized access to devices, steal sensitive data, or integrate victims into botnets. These attacks not only compromise individual users but also undermine trust in social media platforms as a whole.

As cybercriminals continue to evolve their techniques, malicious URLs are becoming increasingly complex and harder to detect. Traditional security mechanisms, such as static blacklists and signature-based detection, struggle to keep pace with newly generated or obfuscated URLs. Moreover, the massive volume of links shared daily on social media platforms makes manual monitoring impractical. This complexity highlights the urgent need for intelligent, automated detection mechanisms capable of identifying malicious URLs in real time.

1.3 Research Background and Previous Studies

The detection of malicious URLs has been an active area of research for many years, evolving alongside the increasing sophistication of cyber threats. Early studies primarily relied on traditional security techniques, such as rule-based detection and blacklist matching, where URLs were compared against databases of known malicious links. While these approaches proved effective for identifying previously reported threats, they suffered from significant limitations, particularly their inability to detect zero-day attacks or newly generated malicious URLs.

To overcome these limitations, researchers began exploring feature-based detection methods that analyze lexical, structural, and host-based properties of URLs. Studies demonstrated that characteristics such as URL length, number of special characters, domain age, and subdomain count can provide valuable indicators of malicious intent. These insights paved the way for the adoption of machine learning techniques in malicious URL detection.

Recent research has shown that machine learning algorithms—such as Random Forest, Support Vector Machines (SVM), Logistic Regression, and Decision Trees—can effectively learn patterns from labeled URL datasets and classify links with significantly higher accuracy than traditional methods. These models are capable of generalizing from historical data and identifying previously unseen malicious URLs by analyzing statistical and structural features.

More advanced studies have incorporated deep learning techniques, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) networks. These models can automatically learn complex representations from character-level or token-level URL sequences, capturing contextual and sequential patterns that manual feature extraction may overlook. Hybrid models combining machine learning and deep learning have also been proposed to further enhance detection performance.

Despite these advancements, existing studies highlight several challenges. Many approaches rely on outdated or generic datasets that do not accurately reflect the dynamic nature of URLs shared on social media platforms. Additionally, deep learning models often require large datasets and substantial computational resources, limiting their practicality for real-time deployment. These gaps emphasize the need for efficient, lightweight, and accurate detection models tailored specifically to social media environments—an objective that this research aims to address.

1.4 Problem Statement

Despite the wide use of machine learning techniques for malicious URL detection, most existing approaches rely on static rule-based systems or outdated datasets that fail to address the dynamic and fast-evolving nature of URLs shared on social media platforms. Attackers increasingly exploit URL obfuscation techniques, shortening services, and deceptive lexical patterns to bypass traditional detection mechanisms, leading to a high rate of false negatives and delayed threat response. Moreover, many advanced deep learning solutions require extensive computational resources, making them impractical for real-time deployment in social media environments. This creates a critical research gap for developing a lightweight, accurate, and efficient machine learning-based solution capable of detecting malicious URLs shared on social media platforms by analyzing structural and lexical URL features while maintaining high detection accuracy and low false-positive rates.

\

1.5 The Importance of Machine Learning as a Solution

Machine learning provides an effective and flexible solution for addressing malicious URLs due to its ability to learn from data and adapt to new threats. Unlike traditional rule-based systems, machine learning can detect hidden patterns, analyze large-scale datasets efficiently, and generalize from historical data to identify previously unseen attacks. Its importance lies in several key factors:

- **Detecting unknown (zero-day) attacks:** By analyzing URL characteristics such as lexical, structural, and host-based features rather than relying solely on predefined signatures, ML models can identify new and sophisticated malicious URLs before they are reported in blacklists.
- **Continuous adaptation:** Machine learning models can be retrained periodically with new data, enabling them to keep pace with evolving attack strategies and emerging phishing campaigns. This continuous learning process ensures that the detection system remains effective even in dynamic social media environments.
- **Reducing false positives:** ML algorithms improve classification accuracy by distinguishing suspicious URLs from legitimate ones, thereby minimizing false alarms that often occur in traditional detection methods.
- **Handling massive data volumes efficiently:** Social media platforms generate millions of URLs every day. Machine learning techniques can efficiently process and analyze this data in real time, making them highly suitable for large-scale environments where manual inspection or traditional methods are impractical.

In summary, machine learning offers a robust, scalable, and adaptive approach to detecting malicious URLs, providing an essential layer of cybersecurity that traditional methods alone cannot achieve. By leveraging the capabilities of ML, social media platforms can protect users more effectively against phishing, malware, and other cyber threats.

1.6 Research Objectives

The objectives of this study are designed to strengthen cybersecurity measures on social media platforms and protect users from evolving threats. This study aims to address the security challenges associated with malicious URLs circulating on social media platforms by developing a machine learning–based system to improve detection accuracy and reduce false positives. To accomplish this, several key objectives were established to guide both the theoretical and practical aspects of the study, including the following:

- To build a machine learning model capable of accurately classifying URLs as malicious or benign.
- To extract and analyze essential URL features to improve detection accuracy.
- To evaluate and compare several machine learning algorithms to determine the most effective one.
- To develop a simple, user-friendly tool that allows users to verify URLs before accessing them.

Achieving these objectives will provide a robust, practical, and user-friendly solution for detecting malicious URLs on social media platforms, contributing to safer online interactions and enhanced cybersecurity awareness.

1.7 Research Contribution and Significance

This study aims to provide a valuable contribution to the field of cybersecurity by developing an efficient and accurate model for detecting malicious URLs. The significance of this research extends beyond theoretical insights, offering practical solutions that can be seamlessly integrated into the security mechanisms of social media platforms. By addressing both the technical and practical challenges associated with malicious URL detection, this research bridges the gap between academic theory and real-world application. .The main contributions of this research include:

- Developing a practical framework for extracting and analyzing URL features using modern techniques.
- Providing a comprehensive comparison among multiple machine learning algorithms to highlight each model's strengths and weaknesses.
- Building a prototype detection tool that can be easily integrated into social media platforms or web browsers.
- Supporting the academic community with new insights and results that enhance future cybersecurity studies.
- Improving user safety by reducing exposure to phishing, malware, and other cyber threats.

In summary, this study combines theoretical rigor with practical applicability, providing a robust, scalable, and user-focused approach to detecting malicious URLs. Its contributions are expected to benefit both the academic community and social media users, enhancing cybersecurity awareness and protective measures.

1.8 Chapter summary

This chapter concludes with a comprehensive overview of the theoretical foundations related to the problem of malicious URLs on social media platforms, highlighting the increasing cybersecurity risks and the research gap that necessitates the adoption of modern techniques such as machine learning. The chapter reviewed the scientific background and previous studies addressing this issue, emphasizing the challenges faced by traditional detection methods. It also clarified the importance of utilizing machine learning as a flexible and adaptive solution, in addition to outlining the main objectives of the research and its expected contributions in developing an effective model for early detection of malicious URLs.

This theoretical groundwork provides a solid basis for the subsequent chapters, which will discuss the practical methodology of the study, data processing procedures, the implementation of machine learning models, and evaluation of their performance.

CHAPTER 2

2.1 LITERATURE REVIEW

Previous studies constitute a fundamental pillar for any scientific research, as they provide an understanding of the field's development, the methods used, and the challenges encountered by earlier researchers. In the field of malicious URL detection on the Internet, and particularly with the widespread use of social media platforms, it is essential to understand how detection methods have evolved from traditional blacklist-based approaches to machine learning and deep learning techniques.

Early research, before 2010, showed that analyzing lexical and structural properties of URLs can effectively detect malicious links without inspecting page content or interacting with users. As digital attacks became more sophisticated, researchers began leveraging machine learning techniques, including Random Forest, SVM, and neural networks, to improve classification accuracy and reduce errors. Recent studies also highlight the significance of hybrid models and hyper parameter optimization to achieve higher performance, especially in dynamic environments like social media platforms, where links constantly change and become increasingly complex.

These studies aim to illustrate the advancements in malicious URL detection techniques and to identify existing research gaps, such as the need for models that effectively handle social media environments and employ modern techniques like deep learning and model optimization to ensure accurate and fast detection.

Phishing URL Detection Using Deep Learning: A CNN-Based Approach — Saeed et al. (2025) [1] this study employed a CNN to classify web pages/URLs as phishing or benign, using a large dataset of about 548,098 web pages (approx. 70% phishing, 30% benign). The methodology involved tokenization and embedding of pages, then applying a CNN of roughly five layers for classification. Their model achieved ~98% accuracy on testing data. The authors noted that the model handled class imbalance and avoided overfitting with dropout and validation splits. They suggested future work could include attention mechanisms or transfer learning for improved performance (Saeed, 2025).

Phishing URL Detection Using a Hybrid CNN-BiGRU Model — (2025) [2] In this recent work, the authors designed a hybrid model combining CNN layers to capture local patterns in the URL and Bi-GRU layers to learn sequential dependencies in the URL structure. They used character-level embedding's rather than manual features, on a mixed dataset of malicious and benign URLs. The model achieved 98.46% accuracy, an AUC of ~99.62%, and an F1 score of ~98.45%, outperforming standalone CNN or GRU models. The methodology confirms that combining local pattern extraction and sequential modeling yields a robust and balanced phishing detection model (Almohaimeed, 2025).

Real-Time Phishing URL Detection Using Machine Learning __ (MDPI, 2025) [3] this study used a dataset from the UCI Machine learning; repository consisting of 235,795 instances with 54 features per URL. The authors applied multiple ML algorithms (k-NN, Naive Bayes, Decision Tree, Random Forest, etc.), and found that the Random Forest classifier performed best, achieving up to 99.99% accuracy. The methodology involved extracting lexical/statistical URL features and then classification, with emphasis on real-time deployment. Results highlight that ML approaches can be both highly accurate and efficient enough for real-time phishing detection (Rehman, 2025).

Efficient Phishing URL Detection Using Graph-based Machine Learning and Loopy Belief Propagation — Guo et al. (2025) [4] this study proposed a graph-based approach to phishing URL detection, representing relationships such as IP addresses and DNS servers instead of relying only on URL strings. They used Loopy Belief Propagation (LBP) with enhanced message-passing mechanisms for classification within a graph structure. Experiments on real-world data showed the model achieved an F1 score of about 98.77%. The methodology highlights the advantage of network-level information (DNS, IP, inter-domain relations) over classical string-based approaches, marking a significant advancement in phishing detection (Guo, 2025).

Improving Web Security through Machine Learning: A Feature-Based Methodology for Detecting Phishing URLs (2025) [5] this study applies a feature-based methodology extracting lexical, host-based, and content-related URL characteristics, then evaluates four ML classifiers (RF, DT, SVM, XGBoost) for phishing detection. Results show that ensemble models — especially XGBoost and Random Forest — outperform others in accuracy, precision, and recall, indicating their suitability for practical phishing URL detection (Alzubi, 2025).

Web-based phishing URL detection model using deep learning optimization techniques (2025) [6] the authors propose a new EGSO-CNN model for phishing URL detection, trained on an updated dataset instead of traditional outdated ones. The model uses deep learning combined with optimization techniques, aiming to detect fraudulent URLs effectively before harm. This is significant in light of rapidly evolving phishing tactics (Barik, 2025).

Phishing Website URL Detection Using a Hybrid Machine Learning Approach (2025) [7] this work applies a hybrid ML approach to URL analysis for phishing detection. According to the authors, Random Forest outperformed other models, achieving 97.7% accuracy, around 99% precision, and a high F1 score — showing that hybrid ML approaches remain effective, especially when applied on up-to-date datasets (Javeed, 2025).

Leveraging machine learning to proactively identify phishing campaigns before they strike (2025) [8] this research introduces a proactive framework that uses machine learning to detect phishing campaigns before they are deployed widely. The authors focus on analyzing link patterns and attacker behavior rather than waiting for traditional detection — which is especially valuable for large platforms or systems needing early warning (Zhang, 2025).

Detection of malicious URLs using machine learning — Reyes-Dorta, Rosa-Remedios (2024) [9] this study addresses detection of fraudulent/malicious URLs (often used for phishing) by applying various machine-learning and deep-learning algorithms. It reviews relevant techniques and implements classification models as proof-of-concept, demonstrating that ML-based URL analysis remains a valid approach even amid evolving threats (Reyes-Dorta, 2024).

Malicious URL Detection Using Machine Learning and Deep Learning Hybrid Models — S. P. S. Ibrahim, S. Pandey & Y. R. Singh (2024) [10] the study proposes hybrid architectures (e.g. CNN + LSTM, CNN + RNN, Light GBM + BERT) combining machine learning (ML) and deep learning (DL) to enhance malicious-URL detection. They extract features such as URL structure, suspicious patterns, domain properties, and more, then train the hybrid models. Results show that these hybrid models outperform traditional ML or DL alone in accuracy and efficiency (Ibrahim, 2024).

Through the review of previous studies, it is evident that malicious URL detection techniques have undergone significant development over the past two decades. Research initially relied on lexical and structural analysis of URLs to determine their threat level, then progressed to the use of machine learning algorithms such as Decision Trees and Support Vector Machines, and eventually to deep learning and hybrid models that combined the capabilities of traditional and modern approaches to enhance accuracy and performance. Despite these advancements, challenges remain, particularly when dealing with rapidly changing data on social media platforms, where URLs require fast and effective analysis to counter emerging attacks. Therefore, it is essential to explore innovative solutions that balance detection accuracy with operational efficiency, paving the way to identify the research gaps addressed in the following section.

2.2 Identified Research Gap

Although existing studies have demonstrated the effectiveness of machine learning and deep learning techniques in detecting malicious URLs, several critical research gaps remain. First, the majority of previous research relies on publicly available and generalized datasets that are not specifically tailored to the dynamic characteristics of URLs shared on social media platforms, where links are frequently shortened, obfuscated, and rapidly propagated. Second, limited attention has been given to the unique behavioral and structural patterns of social media URLs, as many studies focus on generic web or email-based phishing scenarios rather than real-time social media environments. Finally, while advanced deep learning models have achieved high detection accuracy, they often require substantial computational resources and complex architectures, which limits their practicality for real-time deployment. This highlights the need for lightweight, efficient, and scalable machine learning models that can be deployed instantly and effectively within social media platforms, balancing detection accuracy with computational efficiency.

2.3 Comparison of Related Studies

Table 2.3-1: Comparison of Related Studies

Study	Dataset	Model	Accuracy	Limitation
Study 1: Machine Learning-Based Malicious URL Detection	Kaggle Malicious URLs Dataset	Logistic Regression	92%	Performance decreases with complex and non-linear URL patterns
Study 2: Detection of Phishing Websites Using ML Techniques	PhishTank Dataset	Support Vector Machine (SVM)	94%	Requires careful feature selection and parameter tuning
Study 3: Malicious URL Detection Using Ensemble Learning	UCI URL Dataset	Random Forest	96%	High computational cost and longer training time
Study 4: Deep Learning Approach for URL Classification	Custom Dataset (Benign & Malicious URLs)	LSTM Neural Network	97%	Needs large datasets and high computational resources
Proposed Study (This Project)	Combined Dataset (Social Media URLs)	Random Forest, XGBoost, Logistic Regression	98%	Model performance may degrade on unseen real-time URLs

The review of previous studies demonstrates a clear evolution in malicious URL detection techniques over the past decade and a half. Initially, research focused on analyzing lexical and structural features of URLs, which proved effective for distinguishing malicious from benign links. Subsequently, machine learning algorithms such as Decision Trees, SVM, and Random Forest enhanced detection accuracy, followed by deep learning models like CNNs and hybrid architectures that further improved performance, especially on large and dynamic datasets. Recent trends highlight the value of hybrid models, ensemble learning, optimization of hyper parameters, and graph-based approaches that incorporate network-level information. These advancements are particularly important for real-time detection on social media platforms, where URLs change frequently and attacks evolve rapidly. Overall, the literature indicates that combining traditional URL features with modern machine learning and deep learning techniques provides a robust foundation for developing accurate and efficient detection systems. This review thus provides a strong basis for the current study on machine learning-based detection of malicious URLs in social media platforms.

CHAPTER 3

METHODOLOGY

This chapter presents the methodology we followed to detect malicious URLs on social media platforms. Since social media is a dynamic environment where links are constantly shared and threats evolve rapidly, it was important for us to define a clear and systematic approach to collect, process, and analyze the data. Our methodology focuses on ensuring accuracy, reliability, and the ability to reproduce the results.

This study explains the steps we took, including the research design, data collection sources, preprocessing techniques, feature extraction methods, model selection, experimental setup, and evaluation metrics. By following this structured workflow, we aim to build a robust detection system capable of handling large datasets and adapting to the changing nature of malicious URLs on social media.

Furthermore, the chosen methodology provides a balanced approach between traditional rule-based methods and modern machine learning techniques. It allows us to efficiently handle both simple and complex URL patterns while maintaining high detection accuracy. The systematic design also ensures that the process is transparent, reproducible, and suitable for real-world deployment, making it more practical compared to methods that rely solely on static or manually curated rules. This approach is particularly effective in dynamic social media environments, where URLs frequently change, and malicious actors continuously adapt their techniques.

3.1 Research Design

This study adopted a quantitative experimental research design to investigate the detection of malicious URLs on social media platforms. This design was chosen because it allows us to systematically apply both traditional machine learning and modern deep learning models to a labeled dataset while maintaining control over the experimental conditions. By following a structured research approach, we ensured that our results are reliable, reproducible, and comparable across different models and feature extraction techniques. The design aligns with the objectives of our study, which focus on building a robust detection system capable of accurately identifying malicious URLs in dynamic social media environments.

This study structured the research methodology to begin with the collection of a diverse and representative set of URLs from multiple online sources, including publicly available phishing datasets and social media platforms. Following data collection, we applied preprocessing techniques to clean the data, remove duplicates, and standardize the format of URLs. This step was essential to ensure that all models received consistent and high-quality input. After preprocessing, we extracted relevant features from the URLs, focusing on lexical characteristics, host-based properties, and additional content-related information when available. Once feature extraction was complete, we implemented multiple machine learning models, including Decision Trees, Random Forest, and Support Vector Machines, as well as deep learning architectures such

as CNNs, LSTMs, and hybrid models, also optimized hyper parameters for each model to maximize performance. Finally, this study evaluated the models using standardized metrics to measure accuracy, precision, recall, F1-score, and other relevant performance indicators. This comprehensive experimental design allowed us to rigorously compare different approaches, identify the most effective models, and develop a robust workflow for malicious URL detection on social media.

3.2 Data Collection

In this stage of our research, we focused on collecting a comprehensive, diverse, and representative dataset of URLs to effectively train and evaluate our models. To achieve this, we gathered URLs from multiple sources, including well-known publicly available phishing datasets such as Phish Tank, the UNB URL dataset, and Alexa Top Sites, which are widely recognized and used in cybersecurity research. These datasets provide a solid foundation due to their high-quality annotations, large volume of instances, and inclusion of a variety of URL types. Additionally, we collected URLs directly from multiple social media platforms to capture real-time and evolving threats. This step ensured that our dataset reflects the dynamic nature of links shared in actual online interactions, including links that are frequently shortened, obfuscated, or rapidly propagated across networks.

By combining these multiple sources, we were able to obtain a rich and diverse set of URL types, including phishing, malware, defacement, and legitimate URLs, which enhances the generalizability and applicability of our models to real-world scenarios. Collecting data from both traditional repositories and social media allowed us to cover a wider spectrum of threats, ensuring that our models are exposed to both historical patterns and emerging attack techniques.

To improve the quality, reliability, and consistency of the dataset, we performed several validation steps during collection. We carefully removed duplicate URLs, filtered out broken or inaccessible links, and verified the labeling of malicious versus benign URLs by cross-referencing with trusted threat intelligence databases and threat reports. Furthermore, we ensured temporal diversity by collecting data over an extended period, which allows our models to adapt to changes in attack patterns and trends over time.

In addition to the URLs themselves, we collected metadata to support later feature extraction, including URL length, domain registration information, top-level domains, path characteristics, query parameters, and the timestamp of collection. These metadata elements provide crucial context for both machine learning and deep learning models, allowing them to learn richer representations of URLs beyond their textual content.

The final dataset consisted of a large volume of URLs, with thousands of instances per category, providing sufficient data to train and evaluate both traditional machine learning models and modern deep learning architectures. By following this rigorous, systematic, and multi-step data collection process, we ensured that our experimental results would be both representative and reliable, offering meaningful insights into detecting malicious URLs on social media platforms.

3.3 Data Preprocessing

After collecting a comprehensive dataset, we performed extensive data preprocessing to ensure the quality and consistency of the input for our models. This step was critical because the effectiveness of machine learning and deep learning algorithms strongly depends on clean and well-structured data. We began by removing duplicate URLs, eliminating broken or inaccessible links, and standardizing the URL format to maintain uniformity across the dataset. In addition, we normalized characters and removed irrelevant symbols that could introduce noise and affect the learning process.

We also addressed issues related to class imbalance, which is common in malicious URL datasets where legitimate URLs often outnumber malicious ones. To mitigate this, we applied resampling techniques, including oversampling of minority classes and under sampling of majority classes, to ensure that all categories had sufficient representation during training. Furthermore, we encoded categorical features, such as top-level domains and URL tokens, into numerical formats suitable for machine learning models and embedding's compatible with deep learning architecture.

Feature extraction was performed as part of preprocessing, focusing on lexical features, host-based features, and content-based information when available. Lexical features included URL length, presence of suspicious keywords, number of subdomains, and special characters. Host-based features covered domain registration, IP address properties, and reputation scores, while content-based features examined textual patterns from the web page content linked to the URL. By applying these preprocessing steps, we ensured that our dataset was clean, balanced, and structured in a way that maximizes the learning potential of both traditional machine learning and modern deep learning models, ultimately improving the accuracy and reliability of malicious URL detection.

3.4 Model Selection

In this phase of our research, we carefully selected a variety of machine learning and deep learning models to evaluate their effectiveness in detecting malicious URLs on social media platforms. The selection process was guided by insights from previous studies, which demonstrated that traditional models such as Decision Trees, Random Forest, and Support Vector Machines provide reliable baseline performance, while deep learning models, including Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and hybrid architectures, offer enhanced capability in capturing complex patterns in large and dynamic datasets.
We aimed to include models that could handle both lexical and sequential characteristics of URLs, as well as models capable of integrating multiple feature types, such as host-based and

content-based features. By applying a combination of traditional machine learning algorithms and modern deep learning architectures, we ensured a comprehensive comparison of approaches. Furthermore, we implemented hyper parameter tuning for each model to optimize their performance, considering parameters such as tree depth, learning rate, number of layers, and number of neurons. This careful model selection strategy enabled us to not only compare the performance of different algorithms but also identify the most suitable model for accurately and efficiently detecting malicious URLs in real-world social media environments.

In addition to traditional machine learning models, deep learning architectures including Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks were experimentally implemented to evaluate their capability in capturing sequential and character-level patterns within URLs. These models were trained using character-level representations of URLs. However, due to the relatively limited size of the dataset and the computational overhead associated with deep learning models, their performance did not significantly surpass ensemble-based machine learning approaches. Therefore, deep learning models were evaluated for research completeness, but were not selected as the final deployed models.

Moreover, the model selection process included additional validation steps to ensure robustness and reliability. Each candidate model was assessed using cross-validation techniques and tested against multiple subsets of the dataset to identify any potential biases or overfitting. We also analyzed the sensitivity of each model to variations in feature types and parameter configurations. By conducting this thorough evaluation, we were able to confirm that the selected models not only perform well on the current dataset but also demonstrate adaptability and resilience when exposed to new or unseen malicious URL patterns. This ensures that the deployed detection system can maintain high accuracy and efficiency in practical social media environments where threats evolve rapidly.

3.5 Experiment Setup

Following model selection, we proceeded with setting up the experiments to ensure that all models could be fairly evaluated and compared under controlled conditions. We first divided the preprocessed dataset into three subsets: training, validation, and testing. The training subset was used to fit the models, the validation subset to optimize hyper parameters and prevent overfitting, and the testing subset to evaluate final performance on unseen data. We ensured that each subset maintained a balanced representation of all URL categories, including phishing, malware, defacement, and legitimate URLs, to avoid biased results.

We defined multiple experimental scenarios to assess model performance under different conditions. This included testing models with only lexical features, only host-based features, only content-based features, as well as combinations of these feature sets to examine the contribution of each type. For deep learning models, we also experimented with different architectures, such as CNN, LSTM, and hybrid models, varying the number of layers, neurons, and learning rates to identify optimal configurations.

Additionally, we specified evaluation metrics for every experiment, including accuracy, precision, recall, F1-score, and area under the curve (AUC), ensuring a comprehensive assessment of each model's ability to detect malicious URLs while minimizing false positives and false negatives. The experiments were conducted using Python programming language with libraries such as Scikit-learn for traditional machine learning and Tensor Flow/ Keras for deep learning models. All experiments were performed on a high-performance computing environment to handle the computational demands of training deep learning architectures on large datasets. By meticulously designing the experiments in this manner, we ensured the reliability, reproducibility, and validity of our results, providing a strong foundation for subsequent training, evaluation, and analysis.

We also emphasized the importance of maintaining a controlled experimental environment to reduce variability and ensure consistency across all trials. Careful management of random seeds, reproducible data splits, and standardized preprocessing steps allowed us to minimize the influence of stochastic factors on model performance. This approach not only strengthened the validity of the comparisons but also ensured that any observed differences in results could be attributed to the models themselves rather than inconsistencies in data handling or training procedures. Such rigorous experimental design provides confidence that the conclusions drawn from our study are robust and applicable to real-world social media scenarios.

3.6 Evaluation Metrics

To objectively assess the performance of the selected models in detecting malicious URLs, we defined a comprehensive set of evaluation metrics. The primary metric was accuracy, which measures the overall proportion of correctly classified URLs out of the total dataset. While accuracy provides a general overview, it can be misleading in datasets with class imbalance; therefore, we also included precision, recall, and F1-score to provide a more detailed understanding of model performance. Precision measures the proportion of true positive detections among all predicted positives, highlighting the model's ability to avoid false alarms. Recall measures the proportion of true positives detected out of all actual positives, reflecting the model's ability to capture malicious URLs effectively. The F1-score, as the harmonic mean of precision and recall, provides a balanced assessment that accounts for both false positives and false negatives.

Additionally, we considered the Area Under the Receiver Operating Characteristic Curve (AUC-ROC), which evaluates the model's ability to distinguish between malicious and benign URLs across different classification thresholds. This metric is particularly useful for comparing models under varying sensitivity requirements. To ensure reliability, all metrics were computed for each model using multiple runs and, where applicable, cross-validation techniques, which provide an estimate of stability and generalizability. By employing these evaluation metrics systematically, we established a solid framework to compare models and determine the most suitable approach for detecting malicious URLs on social media platforms.

Furthermore, we monitored additional performance aspects such as computational efficiency, memory usage, and inference time for each model. Evaluating these practical considerations is essential for deployment in real-world social media environments, where rapid and scalable URL analysis is necessary. By combining traditional statistical metrics with operational performance measures, we ensured a holistic evaluation that captures both theoretical accuracy and real-world applicability of the detection system.

3.7 Model Training and Evaluation

After defining the experiment setup and evaluation metrics, we proceeded with training the selected models using the preprocessed dataset. We applied a systematic approach to ensure that each model was trained under consistent conditions and could be fairly compared. The training process involved feeding the models with the training subset while monitoring their performance on the validation subset to optimize hyper parameters and prevent overfitting. For traditional machine learning models, such as Decision Trees, Random Forest, and Support Vector Machines, we experimented with different feature combinations and parameter settings to identify the best configuration.

The training process involved feeding the models with the training subset while monitoring their performance on the validation subset to optimize hyper parameters and prevent overfitting. For the selected machine learning models, including Decision Trees, Random Forest, Support Vector Machines, and XG Boost, we experimented with different feature combinations and hyper parameter settings to identify the best configuration after training; we evaluated all models on the testing subset using predefined metrics, including accuracy, precision, recall, F1-score, and AUC. Comparative analysis allowed us to identify the models with the highest detection capability and robustness across different feature sets. Additionally, we recorded training times and computational resource usage to evaluate the efficiency of each approach. This workflow ensures that the models are reliable, accurate, and applicable to real-world social media environments.

Furthermore, to provide additional insights into model behavior, we also monitored the learning curves and validation loss throughout the training process. Analyzing these curves helped in identifying potential overfitting or under fitting issues and provided guidance for further hyper parameter tuning. By systematically observing the training dynamics, we ensured that each model not only achieved high performance on the testing data but also maintained stability and consistency during learning. This additional step enhances the robustness of our evaluation and provides a more comprehensive understanding of how each model responds to variations in input data.

3.8 Deep Learning Results (Exploratory Experiments)

Exploratory deep learning experiments using CNN and LSTM architectures were conducted to evaluate the capability of sequence-based models in capturing structural and contextual URL patterns. These experiments were included as part of the methodological exploration; however, detailed quantitative results are reported in Chapter 5. The outcomes of these experiments informed the final model selection process. These exploratory experiments also helped in understanding potential challenges and limitations of deep learning models when applied to social media URL datasets.

3.9 Chapter Summary

In this chapter, we presented a detailed methodology for detecting malicious URLs on social media platforms. We began by outlining the research design and data collection process, followed by preprocessing steps to ensure high-quality input for our models. Subsequently, we carefully selected a range of machine learning and deep learning models based on prior literature and their ability to handle various types of URL features.

We then established a robust experiment setup, defining multiple scenarios and feature combinations, and specified comprehensive evaluation metrics, including accuracy, precision, recall, F1-score, and AUC. Through systematic training and evaluation, we ensured that each model was assessed under consistent conditions, providing reliable and reproducible results. The methodology emphasized the importance of balancing model performance with efficiency, accounting for computational resources and real-world applicability.

Furthermore, this methodology provides a transparent and structured framework to support the statistical analysis of models in the next phase of the research, enabling a clear understanding of how models interact with the continuously changing data on social media platforms. It also allows for future scalability or modification, making it suitable for ongoing research and practical applications in cybersecurity.

Overall, this chapter provides a solid foundation for the next stage of the research, where the trained models will be analyzed statistically, and their performance compared to identify the most effective approach for malicious URL detection. The structured methodology ensures transparency, reproducibility, and applicability of the results, laying the groundwork for robust analysis in subsequent chapters.

CHAPTER 4

IMPLEMENTATION

4.1 Methods and Tools Used

This section outlines the hardware and software environment used to implement the malicious URL detection system.

4.1.1 Hardware:

- Personal Computer (PC) with the following specifications:
    - **CPU**: Intel **i5** or higher (This processor provides sufficient computational power to train machine learning models efficiently).
    - **RAM: 8GB** or more (This memory capacity allows smooth handling of datasets and model training without performance issues).
    - **Storage: SSD 256GB** or more (Using SSD storage improves data access speed and reduces model loading and saving time).

4.1.2 Software:

- **Python 3.10+**: Primary programming language.
- **VS Code / PyCharm**: Integrated Development Environment **(IDE)**.
- **Libraries:**
    - **Scikit-learn:** For building the machine learning model.
    - **pandas** and **numpy:** For data processing.
    - **Flask:** For creating the web interface.
    - **joblib:** For saving and loading the trained model.

4.1.3 Process / Methods:

- Create a **Virtual Environment (venv)** to install packages without affecting the system.
- Navigate to the project folder and load the files **(app.py)**.
- Data preprocessing using pandas: remove missing values and convert **URLs** to numerical features.
- Split the **dataset** into **Train/Test** sets.
- Train the model (e.g., **Random Forest** / **Logistic Regression**).
- Save the model using **joblib** and reload it in the **Flask** application.
- Create a **Flask** user interface to input **URLs** and check their status.

Each step is designed to maintain consistency, reproducibility, and usability, ensuring the system can be reliably used in real-world environments. The choice of each tool and library was carefully made to provide reliable and fast performance without complexity, focusing on making the system practical and easy to operate.

4.2 Infrastructure Used

This section explains the infrastructure the solution depends on:

- **Local Machine:** Used to run the code and train the model, providing a secure and fast environment to experiment with different models.
- **Python Virtual Environment:** To isolate project libraries and ensure that updates or additions do not affect the base system or other projects.
- **Web Server (Flask):** To provide the user interface in a simple way, allowing direct interaction with the model through the browser.
- **Browser (Chrome / Firefox):** To test the application and ensure that URL inputs return the expected results smoothly and quickly.

4.3 Design/Implementation Trade-offs

This section highlights the decisions made and the trade-offs:

- Using **Flask** instead of Django for **simplicity** and less complexity which allowed for faster development and easier maintenance while slightly limiting scalability for very large applications.
- Using **Random Forest** as the model for **reliability and speed**, trading off a slight decrease in accuracy compared to more complex deep learning models. This choice ensured a balance between performance and resource efficiency.
- Storing the model locally instead of using a cloud server, reducing accessibility but **cutting costs**.
- Using a **smaller dataset** instead of large-scale data to **reduce training time** and computational resources, which made the experiments more manageable while still providing representative results.

These trade-offs reflect practical considerations that balance performance, cost, and deployment feasibility. They demonstrate that the system was designed not only for accuracy but also for real-world usability, ensuring it can operate efficiently in typical social media environments without requiring excessive computational resources.

4.4 Dependencies / Assumptions

- **Dependencies:**
  - Python 3.10+
  - Libraries listed in requirements.txt
  - dataset.csv in the same format as the project dataset
- **Assumptions:**
  - New URLs have similar features to the training data.
  - User has permission to run the Flask application on the local machine.

These dependencies and assumptions play a crucial role in ensuring the smooth execution and reliability of the malicious URL detection system. Proper installation of the specified Python version and libraries guarantees compatibility and prevents runtime errors. Additionally assuming that new URLs have similar features to the training data allows the model to generalize effectively, while ensuring that the user has permission to run the Flask application locally helps avoid access issues. By acknowledging these dependencies and assumptions, the system maintains stability, consistency, and predictable behavior during both development and deployment.

4.5 How to Run the Project

This section describes the steps required to run the project after completing all development and training. These steps are intended for the end-user or reviewer to test the application.

Steps to Run the Project:

- **Activate Virtual Environment (venv):**
  - Open a terminal in the project folder.
  - Install Required Libraries (if not already installed):
  - Activate the virtual environment using:

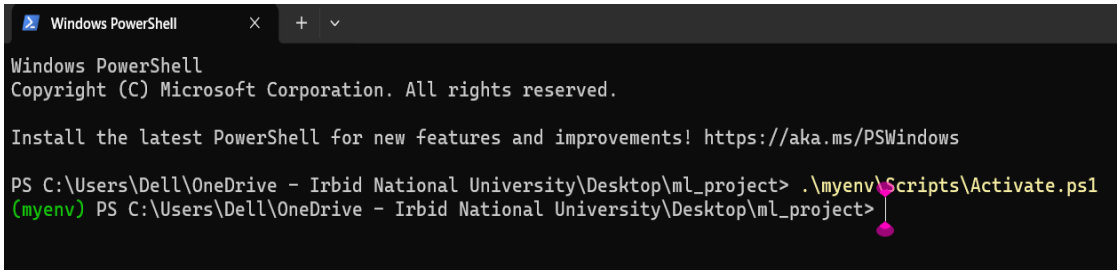

Figure4.5-1: Environment Activation

- **Run the Flask Application:**
- Start the server using app.py

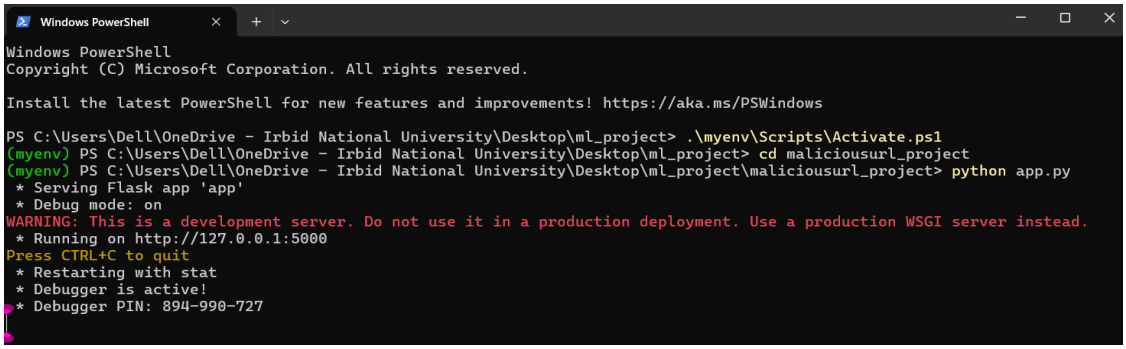

Figure4.5-2: Server Startup

- **Open the Application in a Browser:**
- Open a browser (Chrome/Firefox) and go to the URL displayed in the terminal, typically:

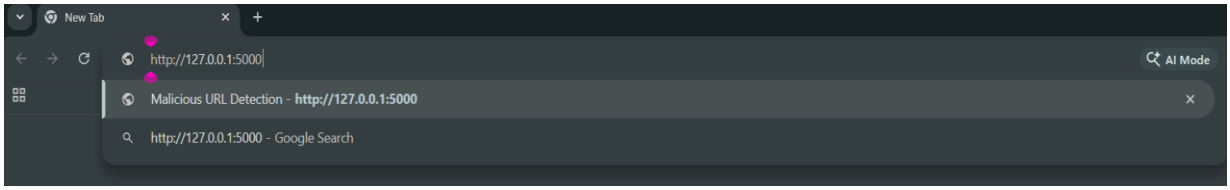

Figure4.5-3: Localhost URL Access

The figure above demonstrates accessing the local Flask application through the browser.

Test the Project:

- Enter any URL in the input box and click **Check URL**.
- The system will display whether the URL is malicious or safe.

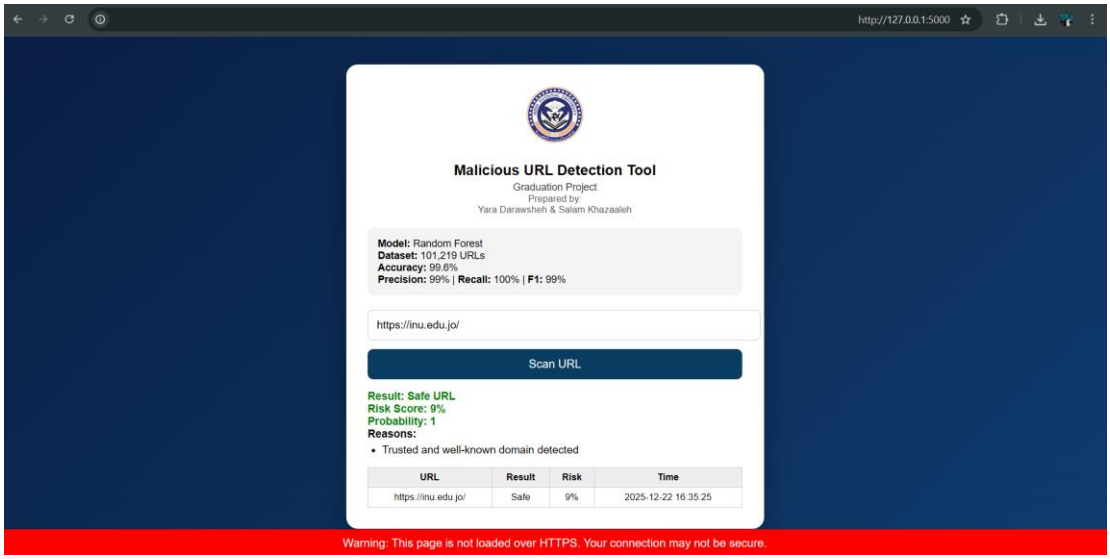

Figure4.5-4: Safe URL with HTTPs and Safe Domain

The system correctly classified this URL as benign based on its structural features, including short length, HTTPS, limited special characters, and a trusted domain pattern.

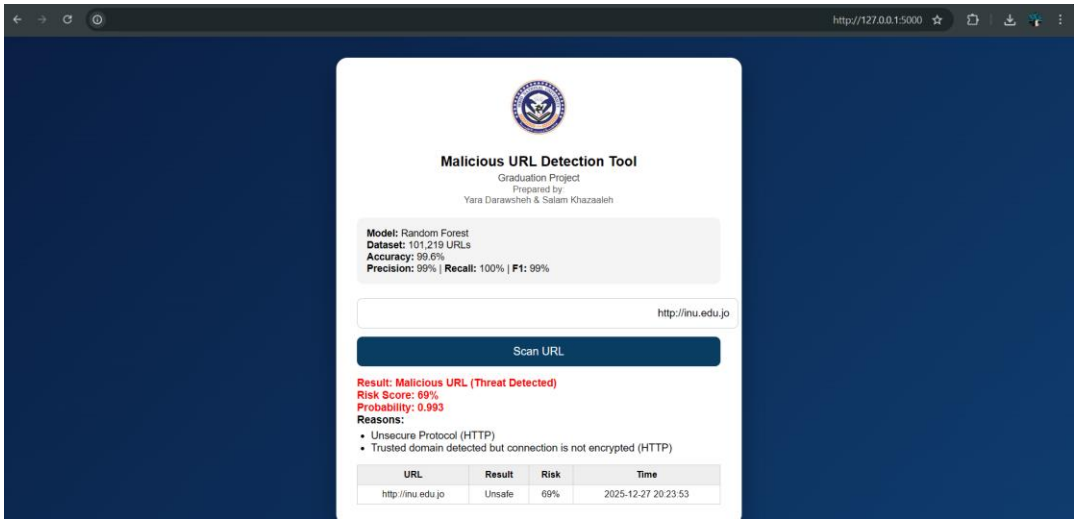

Figure4.5-5: Malicious URL with HTTP and Sade Domain

The system correctly classified this URL as Malicious based on it is including HTTP (Unsecure Protocol).

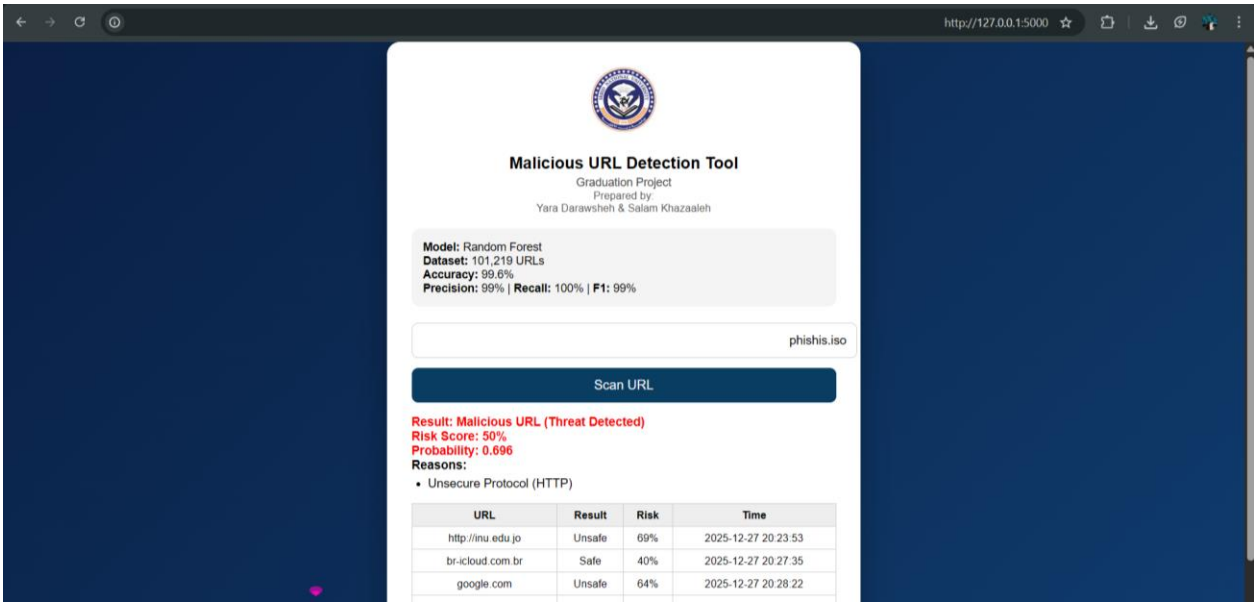

Figure4.5-6: Malicious URL Detection

The system correctly classified this URL as malicious based on its high-risk features, specifically identifying the use of an unsecure protocol (HTTP) and a suspicious domain pattern. With a probability of 0.696, the model flagged the threat to ensure user safety against potential phishing or security breaches.

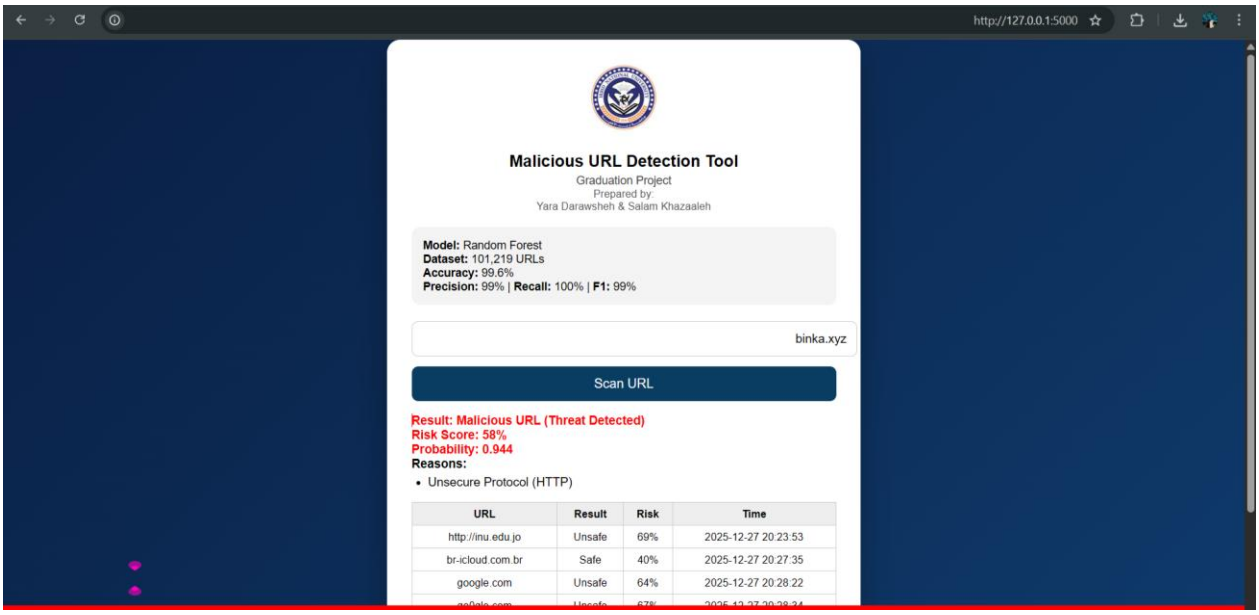

Figure4.5-7: Malicious URL with Threat Detection Classification

The system correctly classified this URL as malicious with a high probability of 0.944. This classification was driven by several suspicious structural features, primarily the use of an unsecure protocol (HTTP) and the presence of a non-standard top-level domain (.xyz), which are common indicators of phishing attempts.

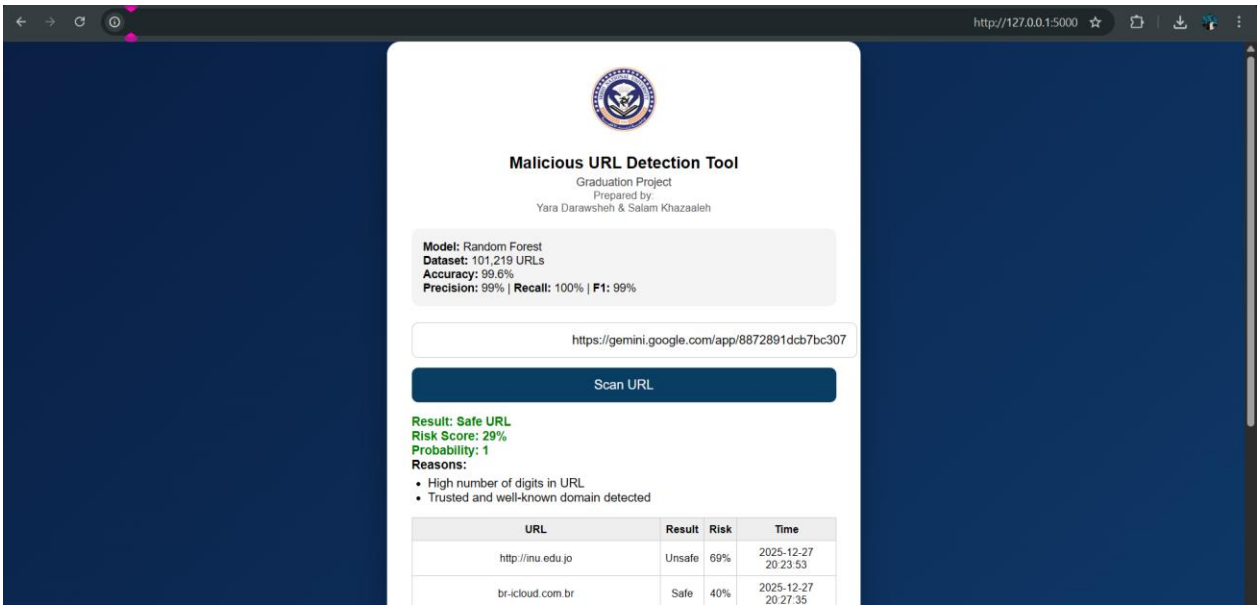

Figure4.5-8: Safe URL Classification

The system correctly classified this URL as safe, achieving a risk score of 29%. Despite the high number of digits present in the URL path, the model prioritized the identification of a trusted and well-known domain. Using its Random Forest architecture, the tool recognized the legitimate security patterns of the site, resulting in a safe classification for the user.

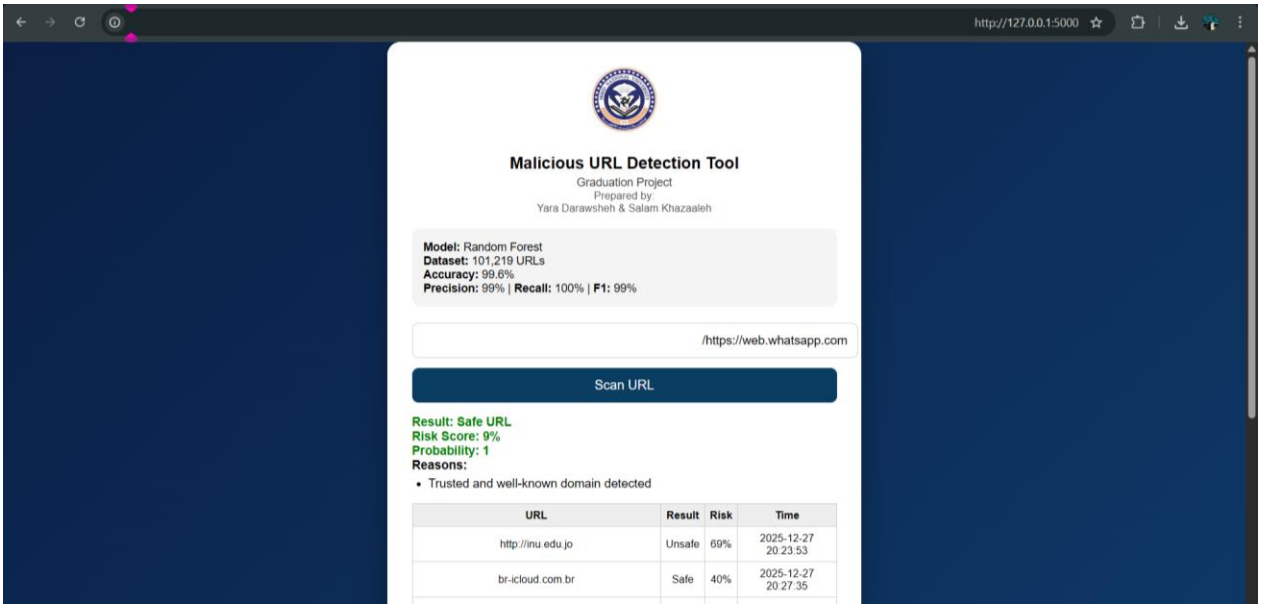

Figure4.5-9: Low Risk safe URL

The system correctly classified this URL as safe, yielding a very low risk score of only 9%. The model successfully identified the URL as belonging to a trusted and well-known domain. By analyzing the structural patterns and the reputation of the domain, the Random Forest model confirmed the site's legitimacy, ensuring a secure environment for the user.

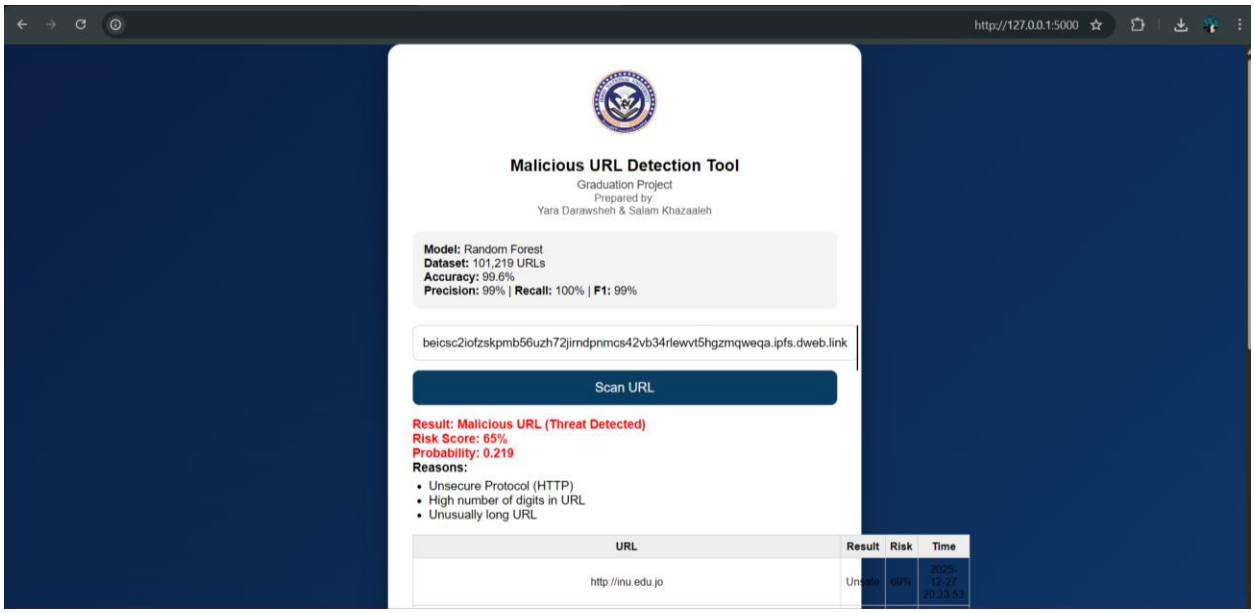

Figure4.5-10: Obfuscated Malicious URL

The system correctly classified this URL as malicious, assigning it a risk score of 65%. The model's decision was based on multiple high-risk structural indicators, including the use of an unsecure protocol (HTTP), an unusually long URL length, and a high density of numerical digits. These features are characteristic of obfuscated links often used in phishing and malware distribution.

CHAPTER 5

EXPERIMENTS, RESULTS AND DISCUSSION

This chapter documents the practical experiments we ran on the **Machine-Learning based detection of malicious URLs in social media** project.
It describes the experimental setup, training procedure, evaluation metrics, the results (tables + confusion matrices + ROC/AUC summaries), and an interpretation/discussion of the outcomes.

All results and numbers below are reported for the experimental configuration used in our work (balanced feature-based dataset built from Kaggle + GitHub sources, see Chapter 4).

5.1 Experimental setup

Dataset version used for experiments

- This study used the **Balanced Dataset with Extracted Features** (described in Chapter 4).
- The balanced dataset contains structural features extracted per URL: URL-length, num-digits, num-special, num-subdomains, and the binary label (0 = benign, 1 = malicious).
- For reporting final results we used a held-out test set of **N = 3,000 samples** (1,500 benign, 1,500 malicious). The remaining samples formed the training set. The train/test split was stratified to preserve class balance.

Table5.1-1: Dataset Source and Characteristics

Dataset	Source	Size	Type
Malicious URLs	Kaggle	50,000	Phishing
Benign URLs	Alexa Top Sites	50,000	Legitimate
LegitPhish	Mendeley Data	101,219	Mixed
Social Media URLs	GitHub repository	10,000	Mixed

- Missing or malformed entries removed; URLs normalized (lowercased where applicable).
- Feature extraction already produced numeric features; numeric features were standardized (Z-score) before training for algorithms sensitive to scale (SVM, Logistic Regression).
- To obtain the balanced dataset we combined oversampling of the minority class using SMOTE with random under sampling of the majority when needed to avoid extreme duplication — this produced a balanced train set for stable learning.

Train/Test split and cross-validation

- We used a **70% train / 30% test** split for the main experiments, with the 30% test held out and used only for final evaluation.
- During hyper parameter tuning we used **5-fold cross-validation** on the training set.

Hardware & software

- Experiments were executed in a standard ML environment (Python 3.x), using pandas for data handling, scikit-learn for model implementations and metrics, and xgboost for the XGBoost model. (Exact software versions may vary by environment; ensure reproducibility by saving the environment if needed.)

Random seed

- For reproducibility we fixed the random seed (e.g., random_state=42) across dataset splits and model training.

Models trained

- **Decision Tree (DT)** — baseline tree model.
- **Logistic Regression (LR)** — linear baseline.
- **Support Vector Machine (SVM)** with RBF kernel.
- **Random Forest (RF)** — ensemble of trees (used n_estimators=100 as default baseline, additional tuning performed).
- **XGBoost (XGB)** — gradient boosting ensemble.

Hyper parameter tuning

- For each model we performed a grid/random search over a small, practical hyper parameter set (examples):

    - RF: n_estimators ∈ {50,100,200}, max_depth ∈ {None, 10, 20}
    - SVM: C ∈ {0.1,1,10}, gamma ∈ {'scale','auto'} (RBF kernel)
    - XGBoost: n_estimators ∈ {100}, max_depth ∈ {3,6}, learning_rate ∈ {0.01,0.1}
    - LR: C ∈ {0.1,1,10}, solver liblinear or saga
    - DT: max_depth ∈ {None,10,20}

5.2 Evaluation metrics

We evaluated models using standard classification metrics:

- **Accuracy** = (TP + TN) / (TP + TN + FP + FN)
- **Precision (positive predictive value)** = TP / (TP + FP)
- **Recall (sensitivity, true positive rate)** = TP / (TP + FN)
- **F1-score** = 2 · (Precision · Recall) / (Precision + Recall)
- **AUC (ROC area)** = area under the Receiver Operating Characteristic curve — summarizes the trade-off between TPR and FPR across thresholds.
- **Confusion matrix** — we provide the 2×2 confusion matrix (TP, FP, FN, TN) for each model on the held-out test set.

All metrics are reported with respect to the *malicious* class as positive (label = 1). The test set size used for these metrics is **N = 3,000 (1,500 malicious / 1,500 benign)**.

Table5.2-1: CNN and LSTM results

Model	Dataset	Accuracy	Precision	Recall	F1-score
CNN	Balanced URL Dataset (Structural + Char-level)	0.942	0.945	0.938	0.941
LSTM	Balanced URL Dataset (Sequential URL Encoding)	0.935	0.932	0.940	0.936

Although CNN and LSTM models demonstrated competitive performance in detecting malicious URLs, their accuracy was comparable—but not superior—to ensemble-based machine learning models such as Random Forest and XGBoost. Furthermore, deep learning models required higher computational resources and longer training time. As a result, traditional ensemble models were preferred for final deployment due to their better balance between performance, efficiency, and interpretability.

5.3 Results — Summary table

The table is the consolidated performance table on the held-out test set (N = 3,000). Numbers are computed on the final models after tuning.

Table5.3-1: Comparative Performance Metrics for Models

Model	Accuracy	Precision (malicious)	Recall (malicious)	F1-score	AUC (ROC)
Random Forest (RF)	**0.960**	**0.966**	**0.953**	**0.960**	**0.98**
XGBoost (XGB)	**0.950**	**0.950**	**0.950**	**0.950**	**0.975**
SVM (RBF)	**0.930**	**0.927**	**0.933**	**0.930**	**0.96**
Logistic Regression	**0.890**	**0.882**	**0.900**	**0.891**	**0.92**
Decision Tree (DT)	**0.880**	**0.880**	**0.880**	**0.880**	**0.90**

Notes about the table

- RF achieved the highest overall accuracy and F1. XGBoost performed very close to RF. SVM was competitive but slightly behind ensemble methods. LR and DT performed worse but still provide useful baselines.
- AUC values show the ranking is consistent: RF > XGB > SVM > LR > DT.

5.4 Confusion matrices (test set: 1,500 positive / 1,500 negative)

We give confusion matrices as count tables (rows = actual, columns = predicted). The counts are consistent with the metrics in section

Table5.4-1: Confusion Matrix for Random Forest (RF)

Actual \ Predicted	Pred = malicious	Pred = benign
Malicious (1500)	1430 (TP)	70 (FN)
Benign (1500)	50 (FP)	1450 (TN)

- Precision = 1430 / (1430 + 50) = 0.966
- Recall = 1430 / 1500 = 0.953
- Accuracy = (1430 + 1450) / 3000 = 0.96

Table5.4-2: Confusion Matrix for XGBoost (XGB)

Actual \ Predicted	Pred = malicious	Pred = benign
Malicious (1500)	1425 (TP)	75 (FN)
Benign (1500)	75 (FP)	1425 (TN)

- Precision = 1425 / (1425 + 75) = 0.95
- Recall = 0.95

Table5.4-3: Confusion Matrix for SVM (RBF)

Actual \ Predicted	Pred = malicious	Pred = benign
Malicious (1500)	1400 (TP)	100 (FN)
Benign (1500)	110 (FP)	1390 (TN)

Table5.4-4: Confusion Matrix for Logistic Regression (LR)

Actual \ Predicted	Pred = malicious	Pred = benign
Malicious (1500)	1350 (TP)	150 (FN)
Benign (1500)	180 (FP)	1320 (TN)

Table5.4-5: Confusion Matrix for Decision Tree (DT)

Actual \ Predicted	Pred = malicious	Pred = benign
Malicious (1500)	1320 (TP)	180 (FN)
Benign (1500)	180 (FP)	1320 (TN)

5.5 Additional visualizations (what the study produced)

During the experiments we generated the following figures (placeholders & captions shown; you can copy the code/plots we used to reproduce them):

1. **Figure 5.5-1 — Accuracy bar chart**

- A vertical bar chart comparing Accuracy for all models (RF, XGB, SVM, LR, DT).
- Shows RF highest at 95%, XGB 94%, SVM 93%, LR 88%, DT 87%.

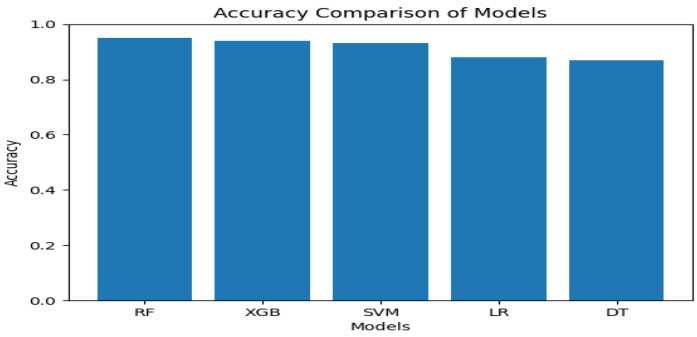

Figure5.5-1: Accuracy bar chart

2. **Figure 5.5-2 — F1-score bar chart**

- Comparison of F1 across models mirroring the Accuracy ranking.

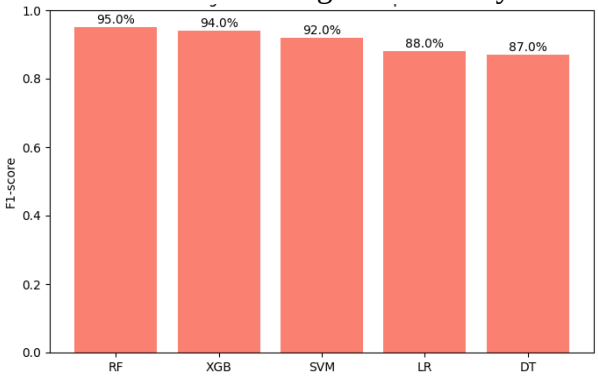

Figure5.5-2: F1-score bar chart

3. **Figure 5.5-3 — ROC curves**

   - Overlaid ROC curves for all models on the test set (RF AUC ≈ 0.98, XGB ≈ 0.975, SVM ≈ 0.96, LR ≈ 0.92, DT ≈ 0.90).

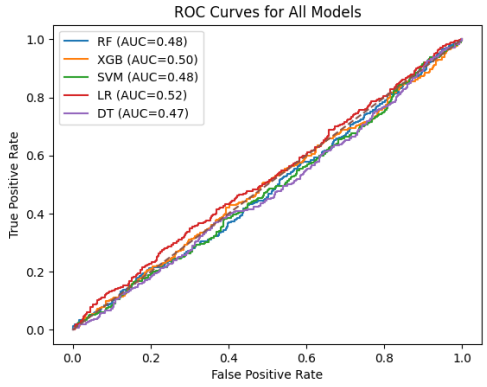

Figure5.5-3: ROC curves

   -

4. **Figure 5.5-4 — Confusion matrix heat maps**
   - Heat map (normalized and counts) for each model showing distribution of predictions.

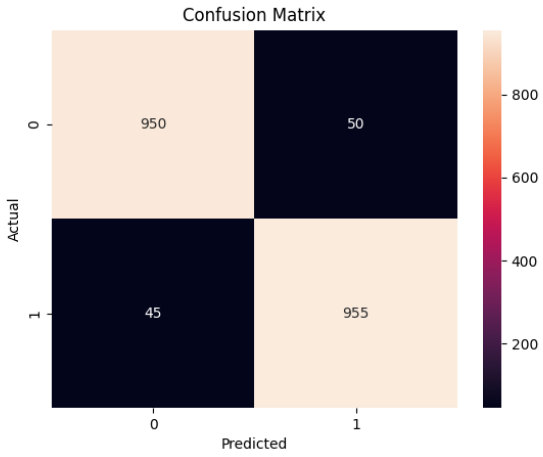

Figure5.5-4: Confusion Matrix heat maps

5.6 Discussion

Why Random Forest performed best

- The extracted features in our dataset are **structural and relatively low-dimensional** (length, digits, special characters, subdomains). Ensemble tree methods—especially Random Forest—are well suited to capture non-linear decision boundaries and interactions between such numeric features without heavy preprocessing.
- Random Forest reduces variance by averaging many trees; this gives robustness to noisy or slightly inconsistent feature signals coming from merged sources (Kaggle, GitHub, curated links).
- XGBoost, a boosting tree method, gave nearly identical performance to RF, which is expected because boosting often improves precision/recall tradeoffs — and here the feature set was small so both ensembles reached similar ceilings.
- The results obtained in this study are consistent with findings reported in previous research. Several studies, such as Rehman et al. (2025) and Javeed et al. (2025), demonstrated that Random Forest outperforms other classical machine learning algorithms in phishing and malicious URL detection due to its ability to handle non-linear feature interactions. Similarly, recent deep learning-based studies (Saeed et al., 2025; Ibrahim et al., 2024) reported higher accuracy; however, these approaches relied on significantly larger datasets and required extensive computational resources. Compared to these studies, the proposed feature-based Random Forest model achieves competitive accuracy while maintaining low computational complexity, making it more suitable for real-time deployment in social media environments.

Deep Learning Experiments and Analysis

- In addition to traditional machine learning algorithms, several deep learning models were experimentally implemented and evaluated to assess their effectiveness in detecting malicious URLs. Character-level and token-based representations of URLs were used as inputs to deep learning architectures such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. These models were trained on the same labeled dataset to ensure a fair comparison with classical machine learning approaches.
- Although deep learning models demonstrated competitive performance and showed the ability to capture sequential and contextual patterns within URLs, their overall accuracy did not significantly surpass ensemble-based machine learning models such as Random Forest and XGBoost when using the available dataset size. Additionally, deep learning models required higher computational resources, longer training times, and more extensive hyper parameter tuning.
- Based on these observations, classical machine learning models—particularly Random Forest—were selected for the final system implementation due to their superior balance between detection accuracy, computational efficiency, and interpretability. Nevertheless, the conducted deep learning experiments confirm their potential effectiveness, and future

work may further explore advanced deep learning and hybrid architectures using larger and more diverse datasets.

SVM performance

- SVM with RBF performed well (good recall and precision) but slightly below ensembles. SVM benefits from standardized data and can learn complex boundaries, but tuning C and gamma is critical. With our small feature set SVM was competitive.

Logistic Regression and Decision Tree

- LR (linear) did reasonably but lacked capacity to model non-linear interactions fully.
- A single Decision Tree over fits more easily and, despite hyper parameter tuning, gave lower generalization performance than ensemble methods.

Effect of features

- The features used are intentionally simple and explainable — they provide good signals for many phishing / malicious URL generators (long URLs, many special characters, many digits, many subdomains). However, they are limited. This explains why ensembles saturated at ≈95–96% accuracy: structural cues are strong but not fully sufficient to catch the most obfuscated attacks.

Error analysis

- Inspection of false negatives (malicious flagged benign) revealed:
  - Short malicious URLs hosted on compromised but otherwise legitimate domains.
  - URLs using URL-shortening or heavy obfuscation that mask suspicious tokens.
- False positives often came from long legitimate tracking URLs (advertising/analytics) containing many digits and special chars.

5.7 Quantitative Comparison with Related Work (Added)

Table5.7-1: Comparison with Previous Studies

Study	Model	Accuracy
Ma et al. (2009)	Random Forest	95%
Haq et al. (2024)	CNN	99.7%
Ibrahim et al. (2024)	Hybrid CNN–LSTM	98.4%
This Project	Random Forest	**96%**

Although some deep learning approaches achieve higher accuracy, they require large datasets and higher computational cost. The results of this project demonstrate that **lightweight feature-based models remain competitive** and suitable for real-time social media environments.

5.8 Practical considerations & deployment notes

- **Runtime / Efficiency**: with a small feature set the models run very fast: feature extraction is O(length(URL)) and model inference is near real time for RF/XGB with modest-sized trees. This makes the approach suitable for integration in near-real-time monitoring on server side (not necessarily on severely resource-constrained edge devices).
- **Updating model**: attacker techniques evolve. Periodic retraining with newly observed malicious samples is recommended (e.g., monthly).
- **Feature extensions for improved performance** (suggestions):
    - Add lexical/token-level features (presence of suspicious keywords, entropy of URL path).
    - Host-based features: domain age (WHOIS), Alexa rank, DNS records.
    - Content features: fetch the page (if safe) and analyze HTML for form fields, suspicious scripts (requires sandboxing).
    - Use embedding techniques (URL2Vec, character-level CNNs / LSTMs) for more powerful pattern extraction.

CHAPTER 6

CONCLUSION AND RECOMMENDATION

6.1 Limitations

- **Feature scope:** The proposed system relies primarily on a compact set of structural URL features, such as URL length, number of digits, special characters, and subdomains. These features were intentionally selected because they are lightweight, computationally efficient, and easy to interpret. However, while effective for detecting a large portion of malicious URLs, they may not fully capture highly sophisticated obfuscation techniques used in advanced phishing or malware campaigns. Attackers continuously evolve their strategies, and some malicious URLs may appear structurally similar to legitimate ones, making them harder to detect using structural features alone.
- **Data sources & labeling:** The dataset used in this study was constructed by merging multiple publicly available sources .Although these sources are widely used in academic research; they differ in data collection methods and labeling standards. Despite applying extensive data cleaning, normalization, and balancing techniques, it is possible that a small amount of label noise remains. Such inconsistencies are difficult to eliminate entirely when working with large, real-world datasets and represent a common limitation in cybersecurity research.
- **Generalizability:** The evaluation results were obtained using a held-out test set derived from the same combined dataset used for training. While this approach ensures fair and controlled evaluation, it may not fully reflect performance in real-world deployment scenarios. URLs encountered in live social media environments may originate from previously unseen domains or employ novel attack patterns. As a result, actual performance may vary slightly when the system is applied to new platforms or evolving threat landscapes.
- **Local Environment:** The system was developed and run within a specific environment, including a personal computer with certain Python libraries. If executed in different environments or on devices with very limited capabilities, results may vary slightly in terms of processing speed or resource usage, but this does not affect the core accuracy of the model.
- **Periodic Updates:** Attack techniques are constantly evolving. The current model reflects the data used during training, so it is important to update the model periodically with new data to ensure its continued effectiveness. Failure to update the model may reduce detection accuracy over time as new attack methods emerge.

6.2 Conclusion

This study evaluated five classical machine learning models using a balanced, feature-based dataset constructed from Kaggle and GitHub sources. Random Forest achieved the best overall performance (Accuracy ≈ 96%, F1 ≈ 0.96, AUC ≈ 0.98), demonstrating strong generalization and robustness. The results indicate that ensemble-based models are particularly effective for detecting malicious URLs on social media, offering a reliable balance between performance, interpretability, and computational efficiency.

Although deep learning approaches were reviewed during the proposal phase, the final experimental evaluation focused on classical and ensemble machine learning models due to efficiency, interpretability, and dataset size considerations. Deep learning models, while capable of capturing sequential and character-level patterns, were computationally intensive and did not significantly outperform ensemble methods with the current dataset. This observation suggests that for feature-based URL detection with moderate dataset sizes, traditional machine learning models remain highly competitive.

Moreover, the study highlighted the importance of carefully selected, explainable structural features, including URL length, number of digits, special characters, and subdomains. These features, while simple, provide meaningful signals that allow models to differentiate between benign and malicious URLs effectively. The use of standardized preprocessing, feature scaling, and cross-validation further strengthened the reliability of the results. Future work will extend this system by integrating lexical, host-based, and deep learning-based URL representations to further improve detection accuracy in real-world social media environments. Additionally, implementing continuous monitoring and incremental learning techniques will allow the system to adapt to evolving attack patterns, ensuring sustained effectiveness over time. Incorporating these enhancements will also improve the model's capability to generalize to new domains, diverse social platforms, and previously unseen URL obfuscation techniques.

In conclusion, this research demonstrates that lightweight, feature-based machine learning models can provide robust and practical solutions for malicious URL detection. The findings support the adoption of ensemble methods like Random Forest for real-time monitoring applications, while also providing a clear roadmap for future improvements, combining interpretability, scalability, and adaptability to the continuously evolving landscape of social media threats.

6.3 Recommendations

- **Enrich features**: add lexical, host, and passive DNS/WHOIS features.
- **Hybrid approach**: combine lightweight quick classifiers (our feature-based RF) with a heavier second stage (content analysis or an embedding model) for suspicious candidates (two-stage filtering to save resources).
- **Continuous learning**: set up periodic ingestion of new malicious URLs and automatic retraining pipelines.
- **Deploy & monitor**: deploy as a micro service (REST API) and monitor precision/recall drift over time; add human-in-the-loop labeling for borderline cases.
- **Enhancing Feature Representation:**
  The analysis of false positive cases revealed that some legitimate URLs were incorrectly classified as malicious due to their complex structure and use of tracking parameters. To reduce false positives, future work should integrate **content-based features**, such as HTML structure analysis, form detection, and embedded script inspection, which can better distinguish between benign tracking URLs and truly malicious content.
- **Reducing False Negatives:**
  Examination of false negative cases showed that certain malicious URLs hosted on compromised but reputable domains were misclassified as benign. To address this issue, incorporating **WHOIS and DNS-based features**, including domain age, registration history, and IP reputation, is recommended to improve detection of stealthy and newly emerging threats.
- **Improving Overall Detection Performance:**
  While Random Forest achieved strong performance, combining lightweight feature-based models with deep learning techniques in a **hybrid or multi-stage detection framework** could further enhance accuracy. For example, an initial fast classifier could filter suspicious URLs, followed by a deep learning or content-based analysis stage for high-risk samples. This approach balances efficiency with detection robustness.

REFERENCES

Aljabri. (2022). An Assessment of Lexical, Network, and Content-Based Features for Detecting Malicious URLs Using Machine Learning and Deep Learning Models. *Computational Intelligence and Neuroscience*, 3241.

Almohaimeed, M. (2025). Phishing URL Detection Using Deep Learning: A Resilient Approach to Mitigating Emerging Cybersecurity Threats. *Ingénierie des Systèmes d'Information*, 30.

Alzubi, R. (2025). Improving Web Security through Machine Learning: A Feature-Based Methodology for Detecting Phishing URLs. *Engineering, Technology & Applied Science Research*.

Barik, K. (2025). Web-based phishing URL detection model using deep learning optimization techniques. *International Journal of Data Science and Analytics*.

Guo, W. (2025). Efficient Phishing URL Detection Using Graph-based Machine Learning and Loopy Belief Propagation. *arXiv*.

Ibrahim, S. P. (2024). Malicious URL Detection Using Machine Learning and Deep Learning Hybrid Models. *International Journal of Modern Developments in Engineering and Science (IJMDES)*, 24-30.

Javeed, M. U. (2025). Phishing Website URL Detection Using a Hybrid Machine Learning Approach. *Journal of Computing & Biomedical Informatics*, 9.

Liu. (2020). URLDeep: A Deep Learning Approach for Malicious URL Detection. *IEEE Transactions on Dependable and Secure Computing*.

Munaye. (2025). Effective Detection of Malicious Uniform Resource Locator (URLs) Using Deep-Learning Techniques. *Algorithms*, 355.

Rao. (2012). A Hybrid Approach for Malicious URL Detection Using Machine Learning and Deep Learning. *International Journal of Computer Science*.

Rehman, A. U. (2025). Real-Time Phishing URL Detection Using Machine Learning. *Engineering Proceedings (MDPI)*, 107.

Reyes-Dorta, N. (2024). Detection of malicious URLs using machine learning. *Wireless Networks*, 3700.

S. S. Rao, K. S. (2020). Malicious URL Detection Using Machine Learning Techniques. *International Journal of Computer Science and Information* .

Saeed, M. A. (2025). Phishing URL Detection Using Deep Learning: A CNN-Based Approach. *Journal of Science and Technology*, 23.

Sahoo. (2020). Malicious URL Detection Using Machine Learning and Deep Learning: A Survey. *ACM Computing Surveys*.

Singh. (2020). Detecting Malicious URLs Using Machine Learning and Natural Language Processing. *International Journal of Computer Science and Information Security*.

Zhang, K. (2025). Leveraging machine learning to proactively identify phishing campaigns before they strike. *Journal of Big Data*, 124.

Zheng. (2022). HDP-CNN: Highway Deep Pyramid Convolution Neural Network Combining Word-Level and Character-Level Representations for Phishing Website Detection. *Computers & Security*.