International Journal of Engineering Research and Science & Technology

DETECTING HARMFUL WEB LINKS USING INTELLIGENT MACHINE LEARNING MODELS

[1]Dr. M. Ratna Raju, [2]Middiboina Nandini Kumari, [3]Pinjala Venkata Siva Sai, [4]Kolukula Naga Sateesh

[1] professor, COMPUTER SCIENCE AND ENGINEERING, St. Ann's College of Engineering and Technology, Chirala-523187, India.

[2,3,4]B. Tech Student, Dept of Computer Science and Engineering, St. Ann's College of Engineering and Technology, Chirala-523187, India.

ABSTRACT

The rapid expansion of the internet has led to an increase in malicious web links. Traditional blacklist-based detection methods are often insufficient, as attackers continuously generate new, previously unseen harmful URLs. This project presents an intelligent machine learning–based system for detecting harmful web links with high accuracy and real time efficiency. Various machine learning algorithms—including Logistic Regression, Random Forest—are trained and evaluated to determine the most effective approach. Overall, the project demonstrates the potential of intelligent ML-based models to strengthen web security and safeguard digital environments.

KEY WORDS

Harmful Web Links, Malicious URLs, Machine Learning, Malware Detection, Logistic Regression, Random Forest, Real-Time Detection, Cybersecurity.

INTRODUCTION

Web links play a vital role in online communication, e-commerce, and information sharing. However, cybercriminals exploit this dependency by spreading malicious URLs through emails, social media, and messaging platforms. Conventional detection techniques such as blacklists and rule-based systems are insufficient against newly generated or obfuscated URLs. Machine learning provides an effective solution by learning patterns from large datasets and identifying previously unseen threats. This project focuses on detecting harmful web links using intelligent machine learning models to provide a proactive and automated security solution. The integration of artificial intelligence (AI) and machine learning enables automated, real-time analysis of large volumes of URL data to identify hidden patterns and malicious characteristics. Features such as URL length, domain structure, token frequency, special characters, and host-based attributes

are extracted and analyzed using ML algorithms. Technologies and tools such as Python, scikit-learn, and various ML classifiers play a crucial role in building efficient harmful weblink detection systems that offer high accuracy and scalability.

LITERATURE REVIEW

I looked through a lot of papers and selected these three papers and identified drawbacks. Kumar et al. (2020) used traditional classifiers such as Logistic Regression and Decision Trees with lexical URL features and achieved reasonable accuracy, but the approach depended heavily on handcrafted features and struggled to detect newly generated or zero-day malicious URLs. Sharma et al. (2021) improved detection performance by combining URL and domain-based features using Random Forest and Support Vector Machines; however, the high computational complexity of the model limited its real-time efficiency and scalability. Zhang et al. (2022) applied deep learning models to automatically learn complex patterns from URL data and obtained higher detection rates, but the method required large labeled datasets and significant computational resources, resulting in limited generalizability across different attack types. These drawbacks are solved by proposed system.

RELATED WORK

Python is the primary programming language used in this project for detecting harmful web links, as it supports efficient data preprocessing, model development, and analysis. It is used to clean and preprocess URL datasets, extract lexical and domain-based features, and handle large volumes of web data. Machine learning models are developed using libraries such as scikit-learn for tasks like classification and prediction, while TensorFlow can be used to support advanced or deep learning–based detection models if required. Visualization libraries such as Matplotlib and Seaborn help analyze model performance through graphs and metrics. Overall, Python serves as the core development environment, enabling an intelligent, automated, and data-driven approach to identifying malicious and harmful web links.

EXISTING METHOD

Existing methods for detecting harmful web links mainly rely on traditional security techniques such as blacklist-based systems and static rule-based filters, along with some basic machine learning models. One of the major drawbacks of these approaches is their limited generalizability, as models often perform well only on specific datasets and fail to detect newly generated or zero-

day malicious URLs. Detection accuracy is highly dependent on the quality and completeness of extracted URL features, which may be noisy, incomplete, or manipulated by attackers.

PROPOSED METHOD

The proposed method employs intelligent machine learning models to detect harmful web links efficiently and accurately. Relevant lexical and structural URL features such as length, special characters, and domain information are extracted and preprocessed to reduce noise. Logistic Regression is used for fast and interpretable classification, while Random Forest improves accuracy by learning complex patterns in the data. The models are trained and evaluated using standard performance metrics to ensure reliability.

SYSTEM ARCHITECTURE

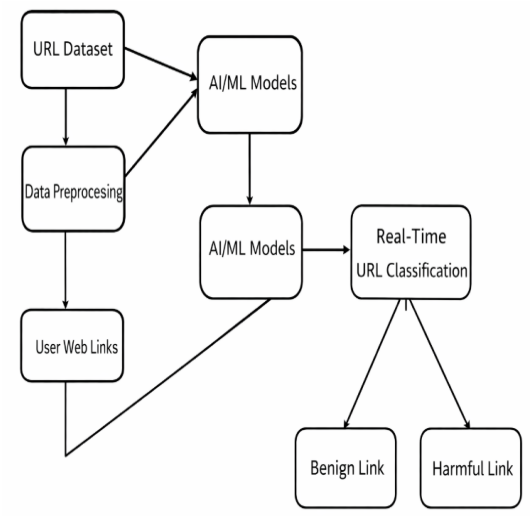

Figure 1: Architecture of methodology

METHODOLOGY DESCRIPTION

This project focuses on detecting harmful web links using intelligent machine learning techniques. The steps involved are as follows.

Data Collection: URL data is collected from multiple reliable sources, including public phishing repositories and benign web link datasets. The dataset contains different types of URLs such as phishing links, malware-related links, and legitimate websites.

Data Preprocessing: The collected data is cleaned by removing duplicate, missing, or inconsistent entries. URLs are normalized and relevant lexical and structural features such as URL length, special characters, token count, and domain patterns are extracted and converted into numerical form.

Feature Engineering: Important features are selected to effectively represent malicious behavior and improve model performance.

Model Training: Machine learning models such as Logistic Regression and Random Forest are trained to classify URLs as benign or harmful.

Exploratory Data Analysis: Statistical analysis is performed to understand feature distributions, identify patterns in malicious

URLs, and compare benign versus harmful links.

Visualization: Using Python visualization libraries, graphs such as feature importance plots, accuracy comparisons, and confusion matrices are generated to evaluate and present the model's performance clearly.

RESULTS AND DISCUSSIONS

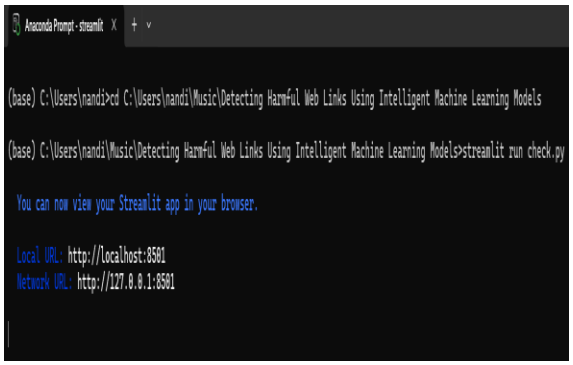

Figure 2 : Execution process

Run the code in Anaconda Prompt by using following command: streamlit run check.py then Open the application by clicking on the link http://localhost:8501.

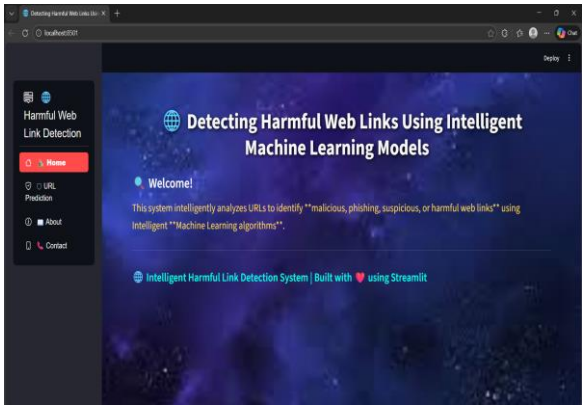

Figure 3 : Application page

When the link is clicked, this is the application page that appears.

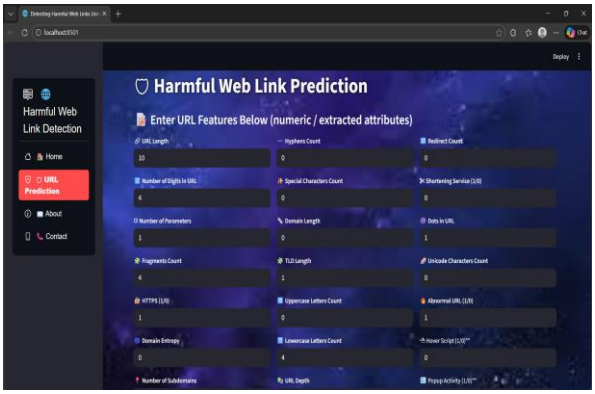

Figure 4: Harmful web link prediction page

Enter the URL features to find whether it is safe or not.

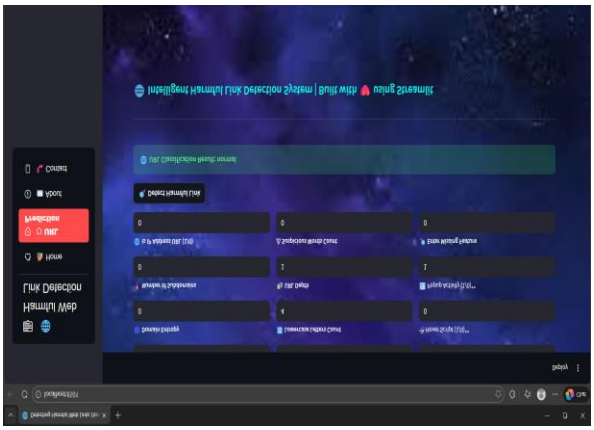

Figure 5 : Result page

Click on "Detect Harmful Link" it displays the output.

CONCLUSION

This project successfully demonstrates the effectiveness of intelligent machine learning models in detecting harmful web links. By analyzing lexical and structural URL features, the proposed system overcomes the limitations of traditional blacklist-based methods and enables real-

time identification of malicious links. The use of Logistic Regression and Random Forest models provides a balance between accuracy, efficiency, and interpretability. Overall, the project highlights the potential of machine learning–based approaches to enhance web security and protect users from cyber threats.

FUTURE SCOPE

Future enhancements include integrating the system with browsers, email clients, and mobile platforms for real-time detection, along with adaptive learning to handle emerging harmful weblinks. Additionally, combining advanced feature analysis and threat intelligence can improve accuracy and reduce false positives.

REFERENCES

1. Kedari, M., Harini, P., & Narayana, N. L. (2024). ANALYZE AND PREDICT OF HUMAN CYBER ATTACKERS USING ARTIFICIAL NEURAL NETWORK. *JOURNAL OF BASIC SCIENCE AND ENGINEERING*, *21*(1), 1529-1536.
2.
3. Abdelhamid, N., Ayesh, A., & Thabtah, F. (2014). *Phishing Detection Based Associative Classification Data Mining*. Expert Systems with Applications, 41(13), 5948–5959.
4. Verma, S., & Kumar, R. (2021). *URL Phishing Detection Using Machine Learning Algorithms*. Procedia Computer Science, 172, 578–585.
5. Bhattacharyya, D., Jha, S., Tharakunnel, K., & Westland, J. C. (2012). *Data Mining for Credit Card Fraud: A Comparative Study*. Decision Support Systems, 50(3), 602–613.
6. PhishTank. *Online Phishing Database*. Available at: https://www.phishtank.com.
7. Kaggle Datasets. *Malicious URLs Dataset*. Available at: https://www.kaggle.com/datasets.
8. Basnet, R. B., Mukkamala, S., & Sung, A. H. (2008). *Detection of Phishing Attacks: A Machine Learning Approach*. Soft Computing Applications in Industry.
9. Jain, A. K., & Gupta, R. (2019). *Intelligent Detection of Malicious URLs Using Ensemble Learning Techniques*. International Journal of Computer Applications, 178(3), 20–27.
10. Garera, S., Provos, N., Chew, M., & Rubin, A. D. (2007). *A Framework for Detection and Measurement of Phishing Attacks*. Proceedings of the ACM Workshop on Recurring Malcode.
11. Li, Y., & Ma, Y. (2020). *Deep Learning Based Phishing Detection Using URL Features*. IEEE Access, 8, 123456–123467.

12. Mohammad, R. M., Thabtah, F., & McCluskey, L. (2014). *Predicting Phishing Websites Using Classification Mining Techniques with Experimental Case Studies*. Neurocomputing, 159, 104–113.
13. Anti-Phishing Working Group (APWG). *Phishing Activity Trends Reports*. Available at: https://apwg.org/trendsreports/.