ShaktiKavach: Realtime Bi modal Browser extension for Phishing Website Detection

Nayan Patil [1], Aryan Pate[2] ,Siddhesh Otavkar[3] ,PrathameshPatil[4]
Department of Information Technology,
Datta Meghe College of Engineering, Airoli, India,

Abstract— **Phishing attacks have become a significant cybersecurity threat, exploiting user trust to steal sensitive information. Traditional detection methods relying solely on URL-based heuristics are often insufficient due to evolving attack strategies. This paper presents a hybrid phishing detection system that integrates URL-based feature extraction and image-based analysis to improve accuracy and robustness. The proposed model employs Random Forest for URL classification using 30 handcrafted features, including HTTPS usage, domain-related attributes, and traffic-based metrics. Additionally, a CNN-based deep learning model analyzes website screenshots to detect phishing attempts based on visual similarities to legitimate sites. A Chrome browser extension is developed to automate URL extraction and real-time classification, enhancing user protection by providing immediate warnings. Experimental results demonstrate that the hybrid approach achieves high detection accuracy, outperforming traditional rule-based methods. This research contributes to advancing phishing detection techniques by combining static and dynamic website characteristics, ensuring faster and more reliable threat mitigation.**

Index Terms—cybersecurity, real time phishing analysis, web security, anti phishing techniques

I. Introduction

Phishing, particularly web phishing, is a major cyberattack that occurs daily, where attackers impersonate legitimate websites to deceive users into revealing sensitive information such as login credentials and payment details. Web phishing attacks have been increasing at an alarming rate. According to the Anti-Phishing Working Group (APWG), an international organization combating phishing, the number of reported attacks grew from 877,536 in Q2 2024 to 932,923 in Q3 2024.

Over the years, various studies have attempted to mitigate phishing attacks using different techniques. However, most approaches focus on URL-based feature analysis or website snapshot-based analysis, limiting their ability to detect sophisticated phishing techniques. Stokes et al. Leveraged URL properties to detect phishing websites, while Md Robiul et al. Used 1D CNN for identification.

To bridge this gap, this study proposes a bi-modal browser extension that leverages both URL features (static, dynamic, and structural) and website snapshots for phishing detection. The key components of this system include:

1. **Bi-modal detection**: Integrating URL-based feature analysis using Random Forest classifier and image-based classification using 2d CNN classifier.

2. **Browser extension**: Developed primarily for Chrome, with compatibility for Brave and Edge.

3. **Parallel processing**: Utilizing Thread Pool to extract 30 URL features simultaneously, optimizing backend processing.

4. **Local storage caching**: Storing safe URLs temporarily to prevent redundant checks, with automatic cleanup after 30 days.

5. **Optimized performance**: Achieving real-time phishing detection within 1-2 seconds.

The rest of this paper is structured as follows. Section II reviews related work on phishing detection techniques, including traditional and machine learning-based approaches. Section III details the methodology, including dataset preparation, feature extraction, and model training. Section IV represents the implementation. Section V presents experimental results and evaluation of the proposed models and implementation. Section VI concludes the paper and outlines future research directions

II. Literature survey

Over the years, various techniques have been developed to tackle phishing attacks. Traditional blacklist and whitelist methods, though widely used, struggle with zero-day phishing attacks and require frequent updates. To overcome these limitations, researchers have explored heuristic-based, machine learning, and deep learning approaches.

Heuristic methods analyze website structures and behavior, while machine learning models use extracted features like domain age, SSL certificates, and traffic patterns for classification. More recently, deep learning techniques, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have enhanced phishing detection by enabling imagebased and text-based analysis.

A. *URL-Based Phishing Detection*

Early phishing detection techniques primarily relied on blacklists. However, these approaches are ineffective against new phishing URLs. Machine learning techniques emerged as an alternative, analyzing URL features to classify phishing and legitimate websites.

Sahoo et al. Conducted a comprehensive survey on malicious URL detection, emphasizing the advantages of machine learning over traditional blacklisting. Islam et al. introduced a 1D CNN model trained on extensive URL features such as URL length, hostname length, and suspicious keywords, achieving 99.85% accuracy. Unlike traditional machine learning models, deep learning techniques demonstrated superior generalization capabilities, effectively detecting zero-day phishing attacks.

B. *Image-Based Phishing Detection*

Since phishing websites often mimic legitimate sites visually, researchers have explored image-based detection techniques. PhishZoo introduced a system that builds visual profiles of trusted websites and detects phishing attempts by comparing page appearances. However, its reliance on stored templates limits adaptability, making it less effective against evolving phishing tactics.

To address this, VisualPhishNet leveraged a triplet CNN to detect phishing sites based on visual similarity. Unlike previous models that relied on direct page matching, VisualPhishNet generalized well to new phishing website designs, making it more robust against evolving threats.

Recent advancements further improve detection by incorporating deep learning-based models, such as CNNs, that analyze visual elements without requiring predefined templates. Our approach enhances these techniques by integrating real-time image analysis within a browser extension, allowing dynamic phishing detection while preserving user privacy.

C. *Our Novelty and Contributions*

Unlike existing phishing detection systems that rely solely on either URL-based heuristics or image-based analysis, our approach integrates both methods within a real-time browser extension, significantly improving detection accuracy. Our system enhances efficiency by implementing parallel processing for faster URL feature extraction, reducing latency in real-time detection.

Additionally, we introduce a local storage caching mechanism to store non-phishing URLs for a defined period, preventing redundant checks and optimizing resource utilization. Moreover, our extension allows user-controlled image-based analysis, ensuring privacy when handling sensitive website screenshots. These enhancements make our solution more adaptive, scalable, and practical for real-world phishing detection.

III. Methodology

A. *Dataset Preparation*

1. Image Dataset

Our image dataset Comprises 400 website screenshots, evenly split between legitimate (200) and phishing (200) websites. Screenshots of trusted websites (e.g., Google, Facebook, Amazon) were manually collected, while phishing website images were extracted from an external dataset.

2. URL Dataset

Our URL Dataset [8] consists of 30+ features having 11,000+ URLS. These URLs are labelled as phishing (1), non-phishing (-1), or neutral (0). The features representing structural and security indicators such as IP usage, HTTPS presence, domain age, redirection behaviour, and more.

B. *Model Training*

1. URL Model

Our Random Forest model analyses different parameters of the URL by extracting features from the url and then these extracted features are trained on multiple machine learning algorithms with a split of 80:20 ratio for training and testing data respectively.

Various models are evaluated using various metrics like Precision, Recall. F1-Score and the roc curve is plotted for visualizing the trend of TPR to FPR. The trained model is stored in pickle(.pkl) format for easy and efficient deployment.

2. Image Model

This model analyses various image parameters using deep learning neural networks. The Convolutional Neural Network (CNN) architecture is employed to train the model. Resizing and Normalization of images are done before training the model and then the model is trained on multiple deep learning classifiers. The dataset is split into 80% training and 20% testing, with the Adam optimizer used for fine-tuning model's performance. The trained model is stored in hdf5(.h5) format for easy and efficient deployment

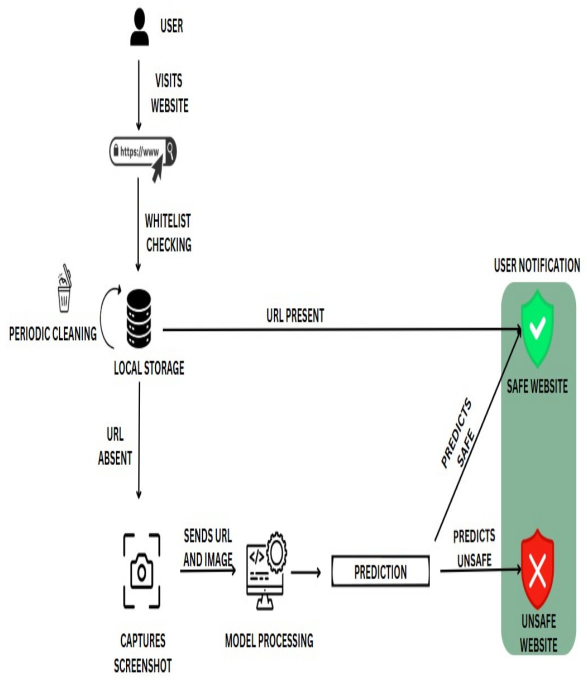


Figure 1 System Architecture

A. System Architecture

1. Frontend: Browser Extension

The browser extension developed using JavaScript, HTML, and CSS, serves as the user interface and real-time monitoring system. It actively captures website data and interacts with the backend for phishing analysis. The key components include:
manifest.json: Defines permissions, enables website monitoring, integrates phishing detection scripts.
background.js: Captures visited URLs and screenshots, sends data to the backend, displays phishing alerts, allows manual whitelisting.

2. Backend: Flask Server

The backend is built using Flask, which processes incoming website data, extracts relevant features, and predicts whether a website is phishing or safe. It consists of three primary modules:
app.py: Manages API routes, calls URL-based and image-based models.
url.py: Extracts 32 features, uses phishing.pkl for predictions, applies multithreading for faster processing.
image.py: Processes webpage screenshots, uses a CNN model (.h5), and classifies sites based on a threshold.

3. Browser Integration Workflow

The extension monitors website activity in real-time, capturing URLs and screenshots when a webpage is loaded or refreshed. Extracted data is sent to a Flask backend API for phishing analysis as represented in Fig.1 . The backend processes the request and send the result to display Based on the API's response, a notification is displayed to warn users if the site is classified as phishing or not as described in Fig. 2.

B. Optimization for Performance Improvement

To enhance efficiency and reduce processing time, several optimizations were applied:

1. **Parallel Processing**: Multi-threading was implemented for faster URL feature extraction, significantly reducing analysis delays.
2. **Local Storage of Safe Websites**: Trusted websites are cached for 30 days to prevent repeated phishing checks on frequently visited sites. Expired entries are removed automatically.
3. **User-Defined Whitelist**: Users can manually mark false phishing alerts as safe, adding the site to a whitelist to improve accuracy and enhance user experience.

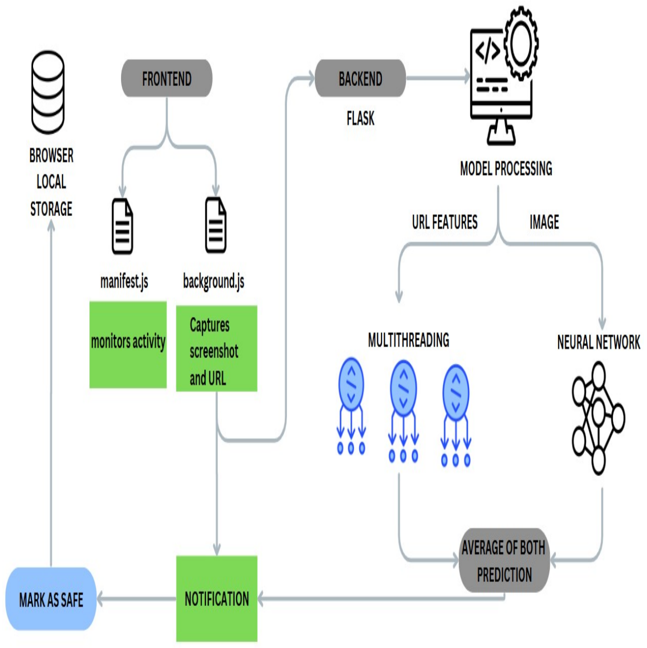


Figure 2 Dataflow

C. Flask API: Integration & Response Handling

The Flask API combines the predictions from both models, calculates a final confidence score, and sends the phishing verdict back to the extension. Various API endpoints are used for smooth and unambiguous data transfer

Table I API Endpoints

Endpoint	Method	Description
/detect	POST	Receives URL and/or screenshot, predicts phishing probability, and returns results.
/check_url	POST	Receives only the URL and returns phishing probability using URL-based features.
/predict_image	POST	Receives only the image, processes it, and predicts phishing probability using deep learning.

IV. RESULTS AND ANALYSIS

A. *URL Model Analysis*

1. Classification Report Analysis

Table II represents the performance of our URL-based phishing detection model using accuracy, precision, recall, and F1-score. The key observations of our model performance are

- **High accuracy** of 96.87%
- F1-score of 0.97 for both classes shows a well-balanced model, maintaining both high precision (avoiding false positives) and high recall (minimizing false negatives).
- Legitimate URLs (0) have a precision of 0.98 but a slightly lower recall (0.95), meaning some legitimate URLs were misclassified as phishing, while phishing URLs (1) have a recall of 0.98, indicating that most phishing sites were correctly detected with minimal false negatives.

2. ROC Curve

Fig 3. Represents the ROC Curve which illustrates the trade-off between true positive rate (TPR) and false positive rate (FPR) across various classification thresholds. AUC score of **99.54%**, indicating a strong ability to differentiate between phishing and legitimate websites.

Table II Classification Report

	Precision	recall	F1-score	support
0	0.98	0.95	0.96	999
1	0.96	0.98	0.97	1234
Accuracy			0.97	2233
Macro avg	0.97	0.97	0.97	2233
Weighted avg	0.97	0.97	0.97	2233

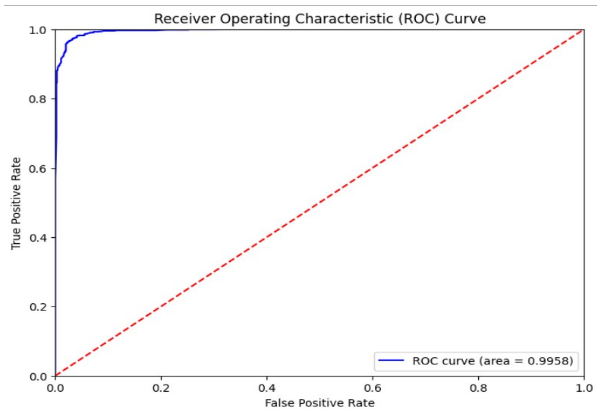


Figure 3 ROC Curve

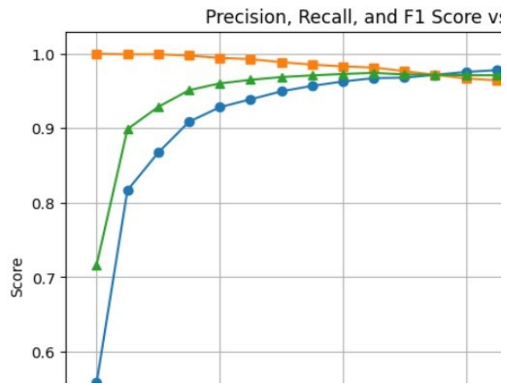


Figure 4 Precision, Recall, F1Score vs Threshold

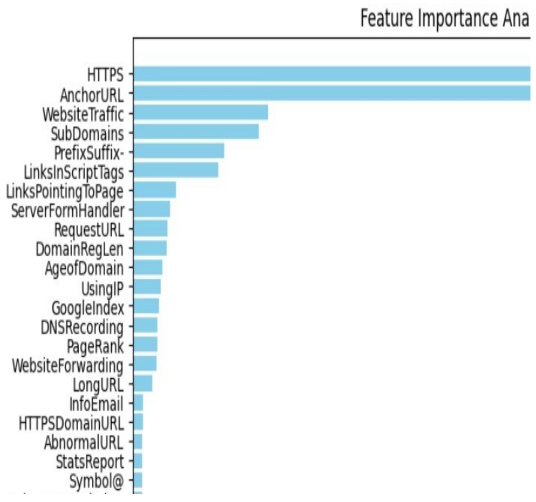


Figure 5 Feature Importance

1. Deciding Threshold

Selecting an optimal classification threshold is crucial for phishing detection, as it directly impacts precision, recall, and the overall F1-score. Different thresholds lead to different trade-offs between these metrics:

- **Precision**: Measures how many of the predicted phishing websites are actually phishing.
- **Recall**: Measures how many actual phishing websites are correctly detected.
- **F1-Score**: The harmonic mean of precision and recall, balancing the trade-off between them.

To determine the best threshold, we analyze how precision, recall, and F1-score vary across different threshold values. With the analysis of Fig 4. The range of optimal threshold comes out to be 0.45-0.55 and we decided 0.55 as it is having a perfect balance of all three parameters

2. Feature Importance Analysis

Understanding key URL features enhances model interpretability. Random Forest classifier assigns importance scores based on how much each feature reduces impurity (Gini index) across decision trees. Fig 5. Represents the importance of features and it shows that **HTTPS** and **AnchorURL** are the most influential factors in phishing detection, indicating that phishing sites often lack HTTPS and manipulate anchor links. **WebsiteTraffic, SubDomains, and Prefix-Suffix** – also contribute significantly, suggesting that low-traffic domains with excessive subdomains and hyphens are common phishing traits. Features like **Favicon, StatusBarCust, and DisableRightClick** have minimal impact, implying that modern phishing sites rarely rely on these deceptive tactics.

3. Comparison with different algorithms

We compared our results which is trained on Random Forest with other algorithms and got the results which are shown in Table III. Based on the comparison, **Random Forest** is selected for our final phishing detection system due to its strong balance of precision, recall, and F1-score, ensuring both **high accuracy and reliability** in identifying phishing websites. The High **recall (0.982)** in **Random Forest** ensures fewer phishing sites go undetected, which is crucial for security applications

Table III Comparision

Sr No.	ML Model	Accuracy	F1score	Recall	Precision
1.	Gradient Boosting	0.974	0.977	0.994	0.986
2.	CatBoost	0.972	0.975	0.994	0.989
3.	XGBBoost	0.969	0.973	0.993	0.984
4.	MLP	0.969	0.973	0.995	0.981
5.	SVM	0.964	0.968	0.980	0.965
6.	Decision Tree	0.960	0.964	0.991	0.993
7.	KNN	0.956	0.961	0.991	0.9989
8.	Logistic Regression	0.934	0.941	0.943	0.927
9.	Naïve Bayes	0.605	0.454	0.292	0.997
10.	**Random Forest**	**0.969**	**0.972**	**0.982**	**0.962**

B. Image Model Analysis

Table IV Summary of proposed CNN Model

Model: Sequential		
Layer(type)	**Output Shape**	**Param**
Conv2D	(None,254,254,16)	448
MaxPooling2d	(None,127,127,16)	0
Conv2D	(None,125,125,32)	4,640
MaxPooling2D	(None,62,62,32)	0
Conv2D	(None,60,60,16)	4,6240
MaxPooling2D	(None,30,30,16)	0
Flattten	(None,14400)	0
Dense	(None,256)	3,686,656
Dense	(None,1)	257
Total params: 3,696,625		
Trainable params:3,696,625		
Non-trainable params:0		

Table V Comparison with Algorithms

Sr No.	Algorithm	Epoch	Accuracy
1.	VGG16	10	92
2.	ResNet50	10	82
3.	MobileNetV2	10	93

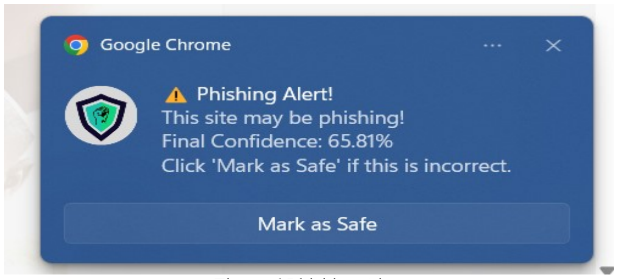
1. Proposed CNN Model analysis

Table IV presents the architecture of the proposed **Convolutional Neural Network (CNN) model** designed for phishing website detection using image-based analysis. The model follows a **sequential architecture**, consisting of three **Conv2D layers**, each followed by **MaxPooling2D layers** for feature extraction and dimensionality reduction. The model also consists of a **total of 3,696,625 trainable parameters**, ensuring sufficient complexity to capture phishing website patterns.

2. Comparison with Different Algorithm

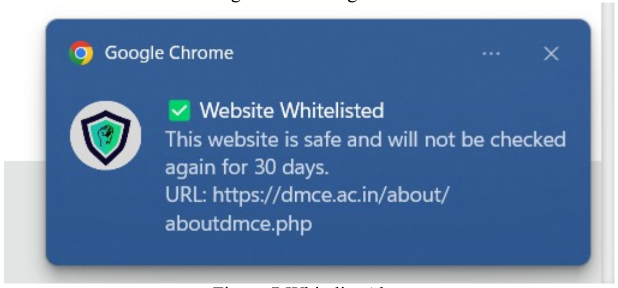
Table V provides a comparison of the proposed model with state-of-the-art **pretrained CNN architectures**, including **VGG16, ResNet50, and MobileNetV2**. The evaluation is based on **10 epochs of training**, and the accuracy achieved by each model is analysed.

C. *Implementation*


Google Chrome

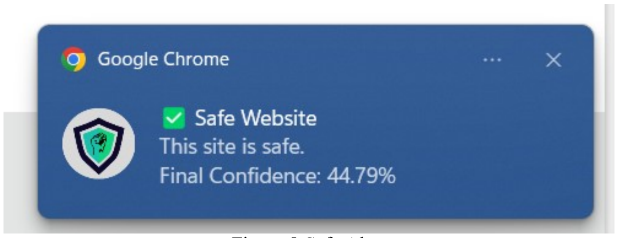
Phishing Alert!
This site may be phishing!
Final Confidence: 65.81%
Click 'Mark as Safe' if this is incorrect.

Figure 6 Phishing Alert


Google Chrome

Website Whitelisted
This website is safe and will not be checked again for 30 days.
URL: https://dmce.ac.in/about/aboutdmce.php

Figure 7 Whitelist Alert


Google Chrome

Safe Website
This site is safe.
Final Confidence: 44.79%

Figure 8 Safe Alert

V. CONCLUSION

Phishing remains a persistent and evolving cybersecurity threat, necessitating robust and adaptive detection mechanisms. In this study, we proposed a hybrid phishing detection approach that integrates URL-based feature extraction with image-based analysis to enhance detection accuracy and resilience against sophisticated attacks. Our system employs a Random Forest model to analyze 30 handcrafted URL features and a CNN-based deep learning model to evaluate website screenshots, effectively capturing both structural and visual phishing indicators.

The propose Chrome browser extension streamlines real-time phishing detection by automating URL extraction, leveraging parallel processing, and implementing local storage caching to optimize performance. Experimental results demonstrate that our hybrid approach outperforms traditional heuristic and rule-based methods, achieving higher accuracy while maintaining low latency.

Future work will focus on expanding the dataset to include more diverse phishing patterns, improving deep learning models with more advanced architectures, and extending support to additional web browsers. By combining static and dynamic phishing detection techniques, our approach offers a scalable and effective solution to mitigate the growing threat of phishing attacks, ultimately enhancing user security in an increasingly digital landscape.

VI. REFERENCES

[1] S. &. G. R. Afroz, "Phishzoo: Detecting phishing websites by looking at them.," *IEEE fifth international conference on semantic computing,* pp. 368-375, 2011.

[2] D. C. L. a. S. C. H. Sahoo, "Malicious URL detection using machine learning: A survey.," *arXiv preprint arXiv:1701.07179,* 2017.

[3] Abdelnabi, Sahar, Katharina Krombholz, and Mario Fritz. "Visualphishnet: Zero-day phishing website detection by visual similarity." Proceedings of the 2020 ACM SIGSAC conference on computer and communications security. 2020

[4] Joshi, Yogesh, et al. "PhishGuard: a browser plug-in for protection from phishing." 2008 2nd International Conference on Internet Multimedia Services Architecture and Applications. IEEE, 2008.

[5] Islam, Md Robiul, et al. "PhishGuard: A Convolutional Neural Network-Based Model for Detecting Phishing URLs with Explainability Analysis." 2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT). IEEE, 2024.

[6] Tajaddodianfar, Farid, Jack W. Stokes, and Arun Gururajan. "Texception: A character/word-level deep learning model for phishing URL detection." ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2020.

[7] https://apwg.org/trendsreports/.

[8] https://www.kaggle.com/datasets/eswarchandt/phishing-website-detector.