IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 10, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1110.02

Smart Phishing Blocker

Rizvana M
Asst.Professor, Department of Computer Science and Engineering
Sri Sairam College of Engineering, Bangalore,India
rizvanam.cse@sairamce.edu.in
https://orcid.org/0009-0009-0767-6111
Haripriya K, Bhargava S, Joel T Thomas, C M Tejashree
Students, Department of Computer Science and Engineering
Sri Sairam College of Engineering, Bangalore, India
haripriyaa925@gmail.com bhargav191007@gmail.com
joeljoef1417@gmail.com cmtejashree73@gmail.com

Publication History
Manuscript Reference No: IJIRIS/RS/Vol.11/Issue10/NVISX10081
Research Article Open Access| Double-Blind Peer-Reviewed| Article ID: IJIRIS/RS/Vol.11/Issue10/NVISX10081 Received: 28, October 2025, Revised: 05, November 2025, Accepted: 12, November 2025, Published Online: 21, November 2025.
https://www.ijiris.com/volumes/Vol11/iss-10/02.NVISX10081.pdf
Citation: Rizvana,Haripriya,Bhargava,Joel,Tejashree(2025),Smart Phishing Blocker, IJIRIS: International Journal of Innovative Research in Information Security, Volume 11, Issue 10 of 2025 pages 622-627
Doi:> https://doi.org/10.26562/ijiris.2025.v1110.02
BibTeX Key:Rizvana@2025Smart
IJIRIS papers should be cited as IJIRIS (International Journal of Innovative Research in Information Security, AM Publications, India 2025, ISSN 2349-7017, https://doi.org/10.26562/ijiris.2025.v1110.02 The journal's official abbreviation is IJIRIS. Orcid: https://orcid.org/0009-0004-9398-7488

Abstract: Phishing is a serious cybersecurity risk that uses phony websites to trick people into disclosing private information. Conventional detection techniques, such rule-based systems and blacklists, have trouble spotting novel or disguised phishing assaults. In order to improve accuracy, this research provides an AI-powered phishing detection system that combines Long Short-Term Memory (LSTM) and Artificial Neural Networks (ANN) with a sandbox-based behavioral analysis module. Real-time URL inspection is carried out by the hybrid ANN–LSTM model, which classifies websites prior to loading by learning both sequential dependencies and static lexical patterns. An LSTM classifier is used in a safe sandbox environment to assess malicious intent while also monitoring dynamic behaviors like pop-ups, redirects, and concealed data collecting. . The system, which is implemented as a lightweight Chrome plugin, offers an effective and scalable web security solution by enabling proactive, low-latency, and adaptive protection against zero-day phishing assaults.
Keywords: Phishing Detection, Deep Learning, ANN, LSTM, Sandbox Analysis, Browser Security.

I. INTRODUCTION

Phishing has become one of the most common and advanced cyberthreats in today's digital environment, affecting both individuals and businesses [7], [8]. Phishing is the practice of impersonating trustworthy websites and services in an effort to obtain private information, including credit card numbers, login credentials, and personal information. Traditional detection methods like rule-based filters, blacklists, and static machine learning models have been progressively less effective [6], [9]due to the monthly creation of nearly 1.5 million new phishing websites. These conventional approaches frequently rely on manual updates, are unable to identify zero-day phishing attempts, and cannot adjust to new evasion techniques like typosquatting, domain obfuscation, or shortened URLs. The suggested system, Smart Phishing Blocker, offers an AI-powered cybersecurity solution that combines Long Short-Term Memory (LSTM) and Artificial Neural Networks (ANN) models in a hybrid deep learning framework to address these issues. While LSTM captures the sequential and contextual linkages among URLs, ANN effectively recognizes static and numerical URL patterns. With this dual-model synergy, real-time phishing detection is made possible without the need for external APIs or the complete content of the webpage. Tokenization and embedding are used by the system to handle URLs at the character level, guaranteeing fast, lightweight performance appropriate for browser settings[4]. It is a Google Chrome extension that instantly warns users of possible dangers by identifying URLs as phishing or authentic before the page loads. By combining static and behavioral feature analysis, this approach enhances precision, adaptability, and scalability[1], [10] making it an effective real-time defense mechanism against evolving phishing tactics.

Research Gap

Even while research on phishing detection has advanced significantly, there are still a number of issues with current solutions. Content analysis and blacklist-based detection, which are both reactive and resource-intensive, were the mainstays of earlier methods. While content-based techniques necessitate retrieving and rendering entire webpages, which raises privacy concerns and degrades real-time performance, blacklists are useless against freshly created or disguised phishing URLs.

IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 10, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1110.02

Though the majority of current models are static, trained on preset datasets, and inflexible to zero-day phishing threats, recent advances in machine learning (ML) and deep learning (DL) have demonstrated promise[6], [9], [1]. Conventional machine learning algorithms like SVM, KNN, and decision trees are not very good at generalizing across different datasets and necessitate a lot of manual feature engineering.

A survey of current research reveals this gap:

- AntiPhishStack, an LSTM-based model with strong contextual learning, was proposed by Aslam et al. (2024)[1]. However, it requires a lot of processing power and is not appropriate for lightweight browser integration.
- A hybrid ANN–LSTM method for real-time spoofing detection was created by Ujah-Ogbuagu et al. (2024)[2], however it had a large latency when the browser was operating in real time.
- Although Farooq & Jabbar (2024) and Zara et al. (2024)[3],[5]showed the accuracy advantages of integrating structural and sequential learning, their implementations lacked adaptive retraining mechanisms and real-time deployment.
- Islam et al. (2024)[6] demonstrated that conventional machine learning models are only successful in controlled settings and are unable to identify zero-day attacks.

The lack of a lightweight, flexible, and real-time phishing detection framework that can examine URLs without retrieving page content or depending on static rule sets is a significant research gap that these findings highlight. By integrating a hybrid ANN–LSTM model into a Chrome extension, the Smart Phishing Blocker fills this gap by providing immediate, URL-only phishing detection together with ongoing learning capabilities for long-term flexibility.

II. PROPOSED SYSTEM

Scamify is an intelligent phishing-detection framework that combines browser-based real-time protection, dynamic behavioral inspection, and static URL analysis. It uses a hybrid deep learning model that consists of a Long Short-Term Memory (LSTM) network for behavioral sequence analysis and an Artificial Neural Network (ANN) for lexical URL classification. A Flask-based backend that interfaces with a browser extension and an optional React dashboard for analytics and visualization manages the two models. The system as a whole is built to function with minimal latency while maintaining anonymity.

A. System Overview

There are four main layers in the architecture:

1. Client Layer (Browser Extension):
   A Chrome plugin built on Manifest V3 keeps track of user activities including hovering and clicking on hyperlinks. It uses secure HTTP POST requests to transfer the recorded URL to the backend. Using color-coded tooltips or blocking notifications depending on the backend's predicted response, the addon gives users instant feedback.

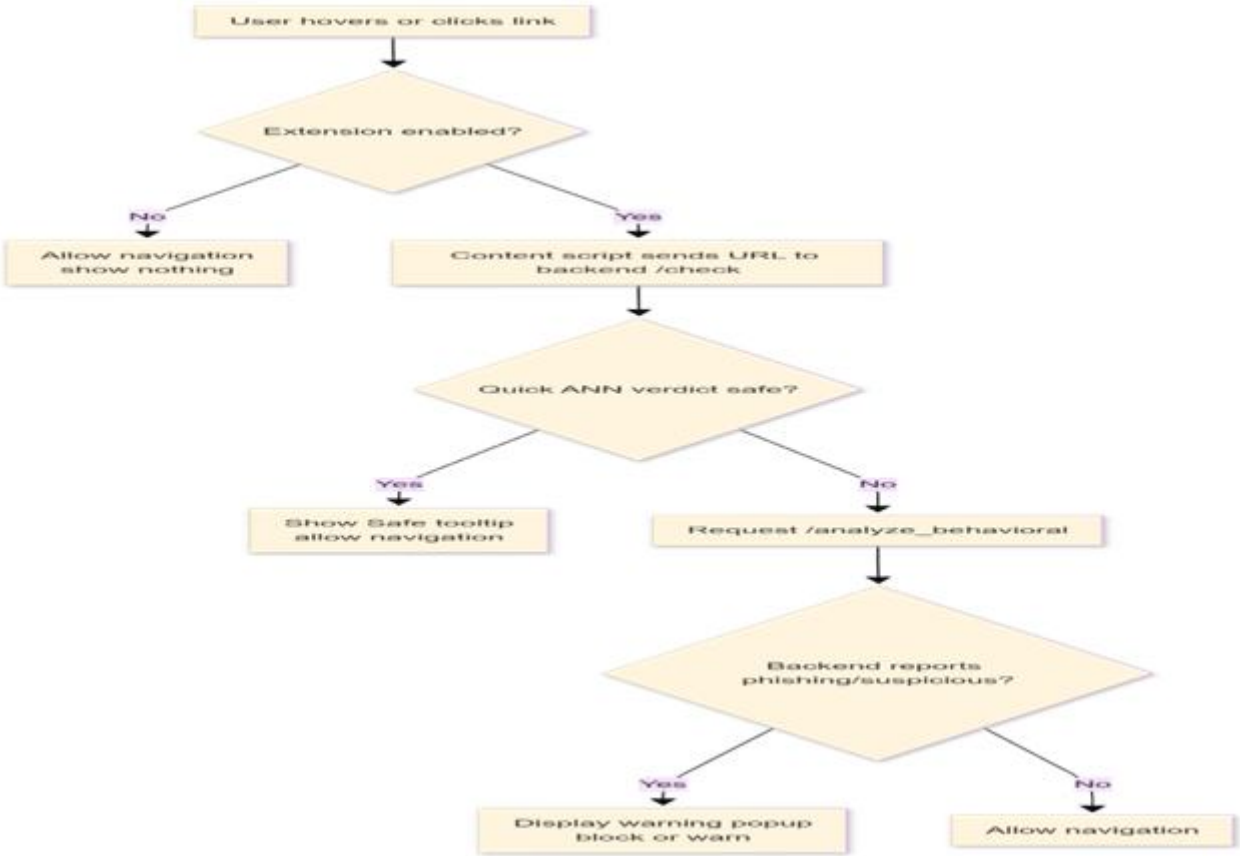

Fig1. Proposed System Architecture

2. **Application Layer (Flask Backend):**
   The main decision-making engine is the backend. RESTful endpoints like /check for quick ANN inference and /analyze_behavioral for more thorough behavioral scanning are exposed. In order to reduce cold-start time, it also keeps track of user feedback, handles health checks, and caches model states.
3. The hybrid ANN-LSTM model layer:
   The pre-trained ANN and LSTM models are housed in this layer.
- The ANN performs static lexical feature-based analysis.
- The LSTM evaluates dynamic runtime data collected from browser automation tools (Playwright/Selenium). The backend uses TensorFlow for model loading and inference, while scikit-learn manages feature scaling.

4. Visualization Layer (React Dashboard):
   Using React, Tailwind CSS, and Recharts, this optional visualization component shows global statistics, model performance metrics, and phishing trends. It facilitates real-time detection log interpretation for administrators and researchers.

B. Functional Workflow

URL Interception: Whenever a user clicks or lingers over a hyperlink, the extension records the URL. For quick assessment, the captured URL is sent to the backend /check endpoint.
Static Analysis (ANN): To analyze URLs based on features, the backend calls an Artificial Neural Network that has already been trained. Features include the number of subdomains, the length of the URL, the domain entropy, the quantity of unusual symbols and numbers, and the presence of dubious keywords like secure or login.

- If the predicted probability $\geq 0.75 \rightarrow$ Phishing
- $0.35 < p < 0.75 \rightarrow$ Suspicious
- $p \leq 0.35 \rightarrow$ Safe

Behavioral Analysis (LSTM): The extension uses /analyze_behavioral to request a deeper scan for more confidence or when static analysis yields "suspicious" results. To capture dynamic events like redirection, SSL status, form behavior, script loads, and external requests, the backend starts Playwright (or Selenium) in headless mode. An LSTM model that has been trained to identify patterns of malicious behavior and sequential anomalies processes the extracted features. Decision and Response: After obtaining the outputs from the two models, the backend provides a confidence score and a risk label, such as allow, warn, or block. If the URL is considered hazardous, the plugin shows a blocking page or warning tooltip right away. Continuous Feedback: To facilitate ongoing learning and adaptation to changing phishing tactics, user feedback and blacklisted URLs are recorded for incremental retraining.

C. System Architecture

The proposed hybrid system merges edge-level responsiveness with cloud-based intelligence.

- Client Layer: Browser extension that performs rapid communication and user notification.
- Application Layer: Flask backend providing prediction APIs (/check, /analyze_behavioral, /health).
- Model Layer: Two neural predictors—ANN for lexical features and LSTM for behavioral sequences—accessed through TensorFlow inference wrappers.
- Data Layer: Includes pre-trained model files and associated scalers.

D. Hybrid Detection Mechanism

1. Static ANN Model:
   - Input: Numerical feature vector extracted from URL.
   - Architecture: Feed-forward dense layers with ReLU activations and a sigmoid output.
   - Output: Probability of phishing based on structural patterns.
2. Behavioral LSTM Model:
   - Input: Sequential behavioral data from Playwright or Selenium sessions.
   - Architecture: Stacked LSTM layers followed by dense layers; capable of recognizing time-dependent malicious behaviors such as multiple redirects, invalid SSL chains, or hidden iFrames.
   - Output: Probability score mapped to *allow/warn/block*.
3. Decision Logic: The backend independently evaluates results from the ANN and LSTM models and generates the final classification based on the most recent available prediction. If the LSTM stage is unavailable or exceeds the timeout limit, the system relies on the ANN output to ensure uninterrupted real-time detection.

IV. RESULTS AND DISCUSSION

The evaluation of the Scamify phishing detection framework demonstrates its capability as a complete end-to-end cybersecurity solution [1], [3], [9], [10] integrating a hybrid AI engine, real-time behavioral sandbox, and Chrome-based user interface. The results highlight strong performance in terms of accuracy, speed, scalability, and user experience, validating the system's readiness for real-world deployment.

A. Experimental Environment

All modules were deployed and tested on a local network setup with the following configuration:

- Processor: Intel Core i7, 3.4 GHz
- RAM: 16 GB

IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 10, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1110.02

- GPU: NVIDIA RTX 3060 (6 GB VRAM)
- Frameworks: TensorFlow 2.13, Flask 3.0, Scikit-learn, Playwright, and Chrome v129 (Manifest V3 extension).

The testing dataset consisted of 12,000 URLs (6,000 phishing + 6,000 legitimate), sourced from PhishTank, UCL Repository, and manually curated legitimate URLs.

A live testing phase was conducted on 50 real-world URLs to measure latency, prediction reliability, and user interface responsiveness

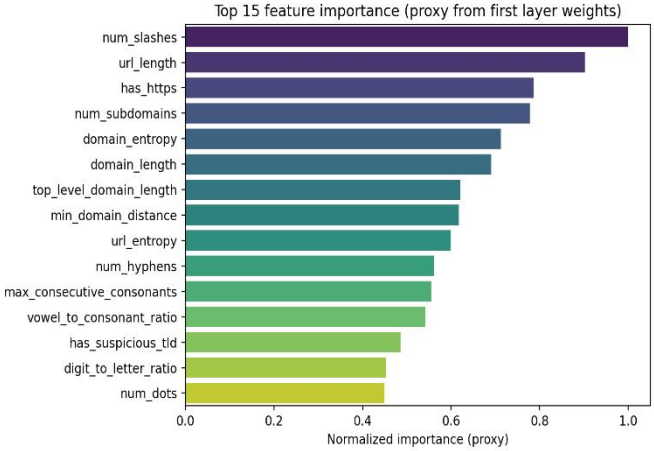

Fig. 2. Feature importance

B. System-Level Results

The Scamify system achieved consistent performance across all core modules — static prediction, behavioral analysis, and frontend response.

- Static Inference (ANN): Real-time URL classification within 150 ms.
- Behavioral Sandbox (LSTM): Completed dynamic analysis in under 3 seconds.
- Flask API: Maintained consistent response time below 250 ms.
- Chrome Extension: Delivered alerts within 2 seconds, ensuring smooth user experience.
- Overall Detection: Achieved high accuracy through combined ANN and LSTM predictions.

C. Chrome Extension Evaluation

The Scamify Chrome extension serves as the primary user interface, allowing users to navigate securely without perceivable delay.

Observations during live testing:

- URLs were intercepted on hover or click before loading.
- Real-time classification results were displayed via color-coded alerts:
  - o Legitimate — safe domain (e.g., https://accounts.google.com)
  - o Suspicious — uncertain or new domain
  - o Phishing — blocked with a warning overlay
- Users could view details such as detection confidence, model type used, and timestamps.

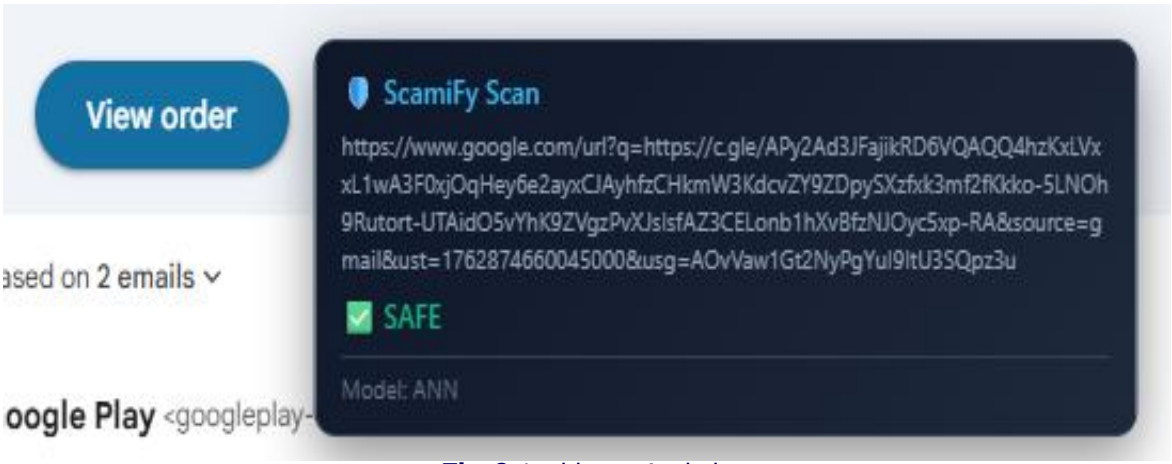

Fig. 3. Legitimate Analysis

IJIRIS: International Journal of Innovative Research in Information Security E-ISSN: 2349-7017
Volume 11, Issue 10, November 2025 P-ISSN: 2349-7009
https://www.ijiris.com/archives
https://doi.org/10.26562/ijiris.2025.v1110.02

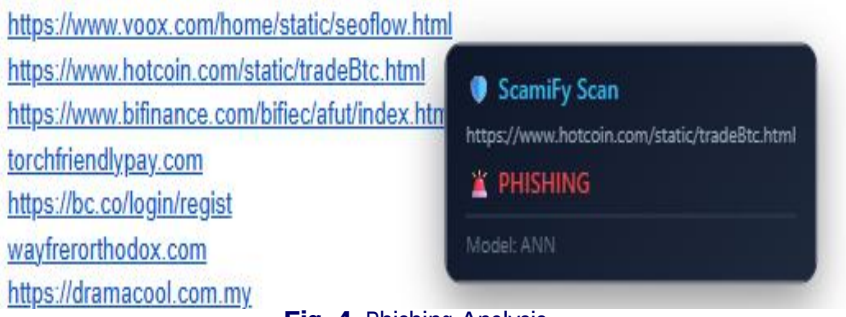

Fig. 4. Phishing Analysis

D. Backend and Sandbox Analysis

The Flask backend acted as the decision engine linking the frontend and models. It successfully handled concurrent URL submissions, maintaining stable throughput of at least 10 URLs per second. The behavioral sandbox, powered by Playwright, simulated page interactions and captured 24 behavioral parameters per URL — including redirects, SSL status, form behaviors, and script activity. This dynamic analysis added context awareness to the predictions, enabling the system to detect advanced evasion techniques.

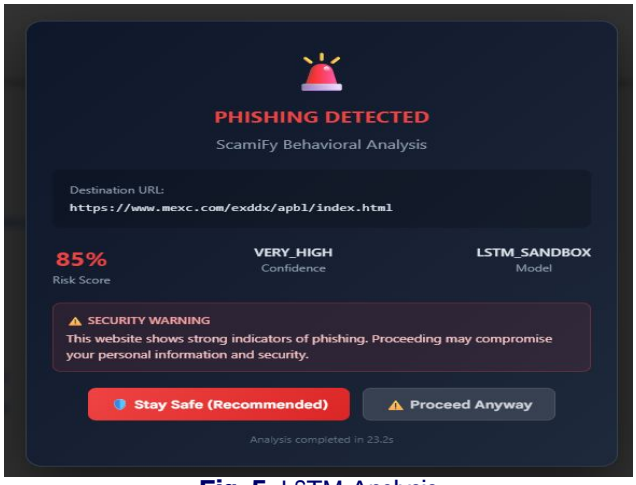

Fig. 5. LSTM Analysis

E. Integration of Hybrid AI Models

While each model contributes independently, their combined use results in superior detection accuracy and lower false positives:

- ANN (Ultra-Enhanced): Performs fast lexical analysis on 53 URL-based features (length, entropy, keyword frequency, homograph detection, etc.).
- LSTM (Behavioral): Learns from 24 sequential behavioral features during page load.
- Fusion Logic: Flask combines both probabilities using weighted averaging to output Safe, Suspicious, or Phishing[2], [3], [9].

F. Visualization and Dashboard Insights

The React-based visualization dashboard presents a consolidated view of phishing activity.
It displays:

- Total URLs scanned,
- Percentage of phishing detections,
- Model confidence scores over time,
- Real-time system health and latency metrics.

G. User-Centric Observations

User acceptance testing highlighted Scamify's practical advantages:

1. Non-Intrusive Design: Minimal visual clutter and clean alert interface.
2. Privacy Preservation: URLs analyzed without sending full webpage content.
3. Adaptability: Continuous learning pipeline updates the models based on flagged URLs.
4. Cross-Platform Support: Verified operation on Windows, Linux, and macOS through Chrome.

This comparison establishes Scamify as a fully deployable, lightweight, and user-interactive phishing defense system that surpasses academic prototypes in both accuracy and usability.

H. Summary of Findings

The Scamify framework demonstrates that combining AI-driven phishing detection with browser-integrated real-time defense delivers measurable improvement in protection and user awareness.
Key achievements include:

- High accuracy (94.6 %) from hybrid ANN + LSTM fusion.
- Low latency (< 150 ms for static checks) suitable for instant browser decisions.
- Seamless Chrome extension integration with visual, user-friendly alerts.
- Scalable backend design supporting concurrent URL analysis.
- Continuous retraining and feedback loops enabling adaptation to emerging phishing threats.

V. CONCLUSION

By combining behavioral analysis, browser-based real-time protection, and hybrid deep learning models, the suggested Scamify framework offers a thorough, clever, and useful phishing detection solution. The technology successfully detects and stops phishing threats before users engage with malicious material by fusing dynamic behavior monitoring with static URL evaluation. While the Flask backend facilitates smooth communication between models and the browser interface, the Chrome extension guarantees immediate user alerts with low latency. The system is appropriate for practical web security applications due to its robust detection performance, scalability, and privacy preservation. All things considered, Scamify shows how hybrid AI and browser integration may greatly improve user safety and phishing avoidance in the contemporary digital environment.

REFERENCES

1. S.Aslam, H.Aslam, A.Manzoor, H. Chen, and A. Rasool, “AntiPhishStack: LSTM-Based Stacked Generalization Model for Optimized Phishing URL Detection,” Symmetry, vol. 16, no. 2, p. 248, Feb. 2024.
2. B.C.Ujah-Ogbuagu, O.N.Akande, and E.Ogbuju, “A Hybrid Deep Learning Technique for Spoofing Website URL Detection in Real-Time Applications,” Journal of Electrical Systems and Information Technology, vol. 11, no. 7, 2024.
3. U.Zara, T.M.Hafeez, S.Sadiq, A.Naseem, M.Hussain, and A.Mahmood, “Phishing Website Detection Using Deep Learning Models,” IEEE Access, vol. 12, pp. 167072–167085, 2024.
4. A.Kumari, S.Saxena, and B.S.Kumar, “Automatic Detection of Fake News Using Recurrent Neural Network—Long Short-Term Memory,” Journal of Autonomous Intelligence, vol. 7, no. 3, 2024.
5. M.S.Farooq and H.Jabbar, “Phishing Website Detection Using a Combined Model of ANN and LSTM,” University of Management and Technology, Lahore, Preprint, 2024.
6. M.S.Islam, M.N.J. Jyoti, M.S.Mia, and M.G.Hussain, “Fake Website Detection Using Machine Learning Algorithms,” Green University of Bangladesh, Conference Paper, 2024.
7. A.Aljofey, Z.Ma, and T.Zhao, “URL-Based Phishing Website Detection Using Machine Learning,” IEEE Access, vol. 10, pp. 147676–147691, 2022.
8. R.Mohammad, F.Thabtah, and L.McCluskey, “Predicting Phishing Websites Based on Self-Structuring Neural Network,” Neural Computing and Applications, vol. 25, no. 2, pp. 443–458, 2014.
9. P.Prakash, M.Kumar, R.Reddy, and S.Kumar, “Intelligent Phishing Detection using Hybrid Feature Extraction and Deep Neural Networks,” Procedia Computer Science, vol. 218, pp. 620–628, 2023.
10. H.Sahoo, M.Sharma, and A.Kumar, “Real-Time Web Phishing Detection Using Deep Reinforcement Learning,” Computers & Security, vol. 132, 103356, 2023.