IRJCS: International Research Journal of Computer Science
Volume 12, Issue 04, April 2025
ISSN: 2393-9842
https://www.irjcs.com/archives

Phishing Detection Using Deep Learning for Web-Browser

Premansh Sunil Pareek, Priya Kumari, Vishnu Somesh
Computer Science and Engineering, Vemana Institute of Technology
Bangalore, Karnataka, India
pareekpremansh@gmail.com, priiyaa2003@gmail.com
vishnusomesh_2021@vemanait.edu.in
Ms.Hajira Begum
Assistant Professor,
Department of Computer Science and Technology
Vemana In stitute of technology, Bangalore, Karnataka, India

Publication History
Manuscript Reference: IRJCS/RS/Vol.12/Issue04/APCS10094 | Research Article | Open Access | Double-Blind Peer Reviewed Article ID: IRJCS/RS/Vol.12/Issue04/APCS10094 Received: 06, April 2025, Revised: 12, April 2025 Accepted: 21 April 2025 Published Online: 05 May 2025 http://www.irjcs.com/volumes/Vol12/iss-04/14.APCS10094.pdf
Article Citation: Premansh,Priya,Vishnu,Ms.Hajira(2025). Phishing Detection Using Deep Learning for Web-Browser**,** IRJCS: International Research Journal of Computer Science, Volume 12, Issue 04 of 2025 pages 195-199
doi:> https://doi.org/10.26562/irjcs.2025.v1204.14
BibTeX Hajira@2025Phishing

Abstract: This paper suggests an end-to-end phishing detection system that includes web and email security hybrid deep learning models. For URL-based threats, URL format and page title are checked by a Char-CNN model, and a multilayer-perceptron deals with structured web attributes. In tandem, the LSTM-based approach is employed for email phishing detection, temporal and contextual extraction patterns through deep sequential learning. Both systems are deployed with scalable RESTful APIs for real-time threat classification. The shared platform emphasizes low latency, model scalability, and real-world integration into current infrastructures.

Keywords: Phishing detection, Char-CNN, LSTM, hybrid model, deep learning, cyber security, REST API, web security.

I. INTRODUCTION

In the digital era, phishing attacks remain one of the most prevalent cyber security threats, deceiving users through deceptive websites that impersonate legitimate platforms. Despite growing awareness and increasing security measures, the sophistication of phishing strategies continues to evolve, necessitating more intelligent and proactive detection mechanisms. Traditional blacklisting methods and shallow heuristics often fall short in detecting novel or obfuscated phishing attempts.[2],[3] This project addresses the problem by introducing a hybrid phishing detection framework that combines deep character-level convolutional neural networks (Char-CNNs) for analyzing textual patterns in URLs and webpage titles, with a machine learning module that processes structured features such as HTML tag frequency, domain entropy, and keyword presence[1],[4].The goal is to capture low-level syntactic errors as well as high-level semantic differences. The solution is architected as a modular design with a Python-based backend that loads the trained hybrid model and classifies incoming web data. The frontend is designed as a minimalist user interface in which people can enter URLs and titles to be validated and get predictions and a confidence score. Scalability of the model, API readiness, and rapid inference are the design focus. This project not only strengthens real-time phishing detection but also lays the groundwork for the creation of future improvement like integration of extensions into browsers, multi-languages support, and adaptive user feedback-based learning.

II. RELATED WORK

Early phishing detection systems relied heavily on manually crafted features and traditional classifiers: lexical and host□based features fed into decision trees, support vector machines, or Naive Bayes models were common throughout the 2010s, but these approaches often struggled when attackers introduced novel obfuscation tactics or dynamically generated content[2],[3],[6]. To address this, researchers began exploring deep learning. Character□level convolutional neural networks (Char□CNNs) were applied to raw URLs and page titles, demonstrating strong ability to learn structural patterns without explicit feature engineering [1]. Parallel work leveraged recurrent architectures—particularly long short□term memory (LSTM) networks—to model the sequential nature of email text, effectively capturing contextual cues such as urgency markers or deceptive phrasing[5],[6]. For example, sequence□to□vector LSTM models have been shown to outperform bag□of□words baselines by preserving word order and long□range dependencies. Hybrid frameworks combining CNNs for structural inputs (URLs, HTML features) with LSTMs for textual content have further improved detection rates, but they often entail complex multimodal pipelines. Our system builds on these advances by focusing exclusively on email bodies and subjects, streamlining the pipeline into a unified LSTM□based model that incorporates rigorous preprocessing HTML stripping, tokenization, and padding to maximize detection accuracy while minimizing latency[4],[7],[8]. By exposing the entire inference workflow via a RESTful API, we achieve both the robustness of deep sequential modeling and the operational simplicity required for real□time email security.

III. SYSTEM DESIGN

A. Design

The phishing detection system is implemented using a monolithic architecture that encapsulates two logically distinct modules within a unified backend service. Each module is dedicated to analyzing a specific type of input: one for detecting phishing in URLs and webpage titles, and the other for detecting phishing in emails. This separation allows for specialized preprocessing and inference logic while maintaining a single deployable service. The URL detection module uses a hybrid deep learning approach combining character-level convolutional neural networks (Char-CNNs) and a multilayer perceptron (MLP) [1],[3],[8]. It takes in a URL, its corresponding page title, and structured features derived from the HTML content. The Char-CNNs processes raw character sequences from the URL and title, while the MLP handles numerical and categorical metadata. The final prediction is obtained after concatenating the feature embeddings from both components. The email detection module is designed around a long short-term memory (LSTM) network, capable of understanding sequential patterns and context in email text [5],[7]. It processes the subject and body of an email, capturing phishing indicators such as urgent language, obfuscated phrases, or embedded phishing URLs. The LSTM output is passed through a dense layer to generate the final classification. Each module is accessible via a dedicated API endpoint exposed through the monolithic backend. The frontend client is responsible for routing user input to the appropriate endpoint based on the selected input type (email or URL).

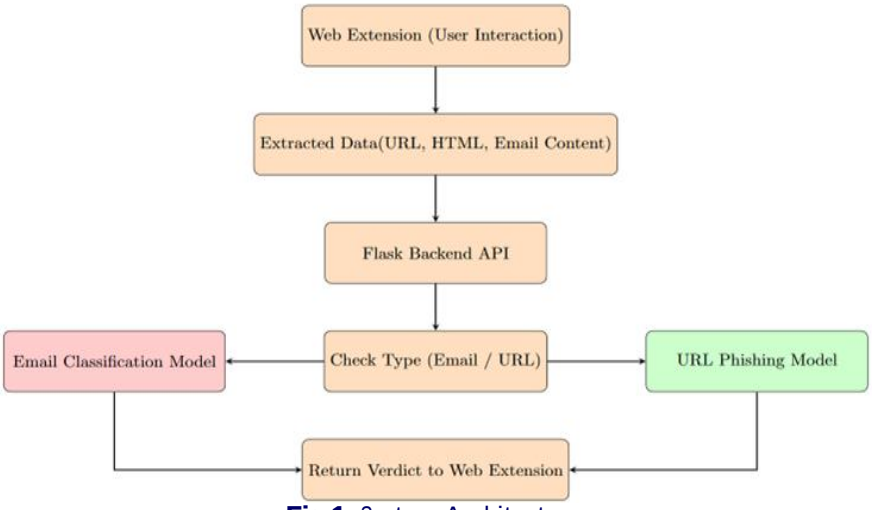

Fig.1. System Architecture

B. System Architecture

The phishing detection system follows a monolithic architecture where both the URL detection and email detection modules are encapsulated within a single backend application. Even though they are housed in the same deployable unit, the two modules are logically distinct and provide distinct API endpoints, one for URL inspection and webpage title inspection, and the other specifically designed to deal email content. This organization facilitates deployment and enables clean functional separation between classes of input. At the user level, interaction is begun through a web-based frontend. The interface assists users to input either a suspicious URL and URL title, or subject and body of an email. Depending on the chosen input type, the frontend sends a POST request to the corresponding backend endpoint. The user interface will be made simple and responsive to give immediate feedback to the users on whether the material submitted is of the phishing variety or original. Once the input is received, the backend determines the appropriate processing path. If the request targets the URL detection endpoint, the input is routed to the hybrid Char-CNN and MLP model. Here, the URL and title are passed through character-level convolutional neural networks to extract lexical and syntactic patterns, while structured features such as domain entropy and tag counts are processed via a multilayer perceptron. The resulting embeddings from all three branches are concatenated and passed through a dense layer to produce a binary classification.
If the request is for email detection, the backend forwards the subject and body text to a long short-term memory (LSTM) model. The email content is first tokenized and embedded, and then passed through stacked LSTM layers to capture sequential and contextual dependencies. The final hidden state is processed by a dense layer to generate the phishing prediction. This module is particularly effective in detecting language-based deception and patterns commonly found in phishing emails. All inference operations are stateless and executed in real time. The backend returns a JSON response that includes the classification label and a confidence score. Throughout the process, both modules operate independently but share the same runtime environment, ensuring a streamlined and scalable detection framework within a unified service. The core of the email phishing detection system revolves around an LSTM□based sequential model designed to capture temporal and contextual cues within message content. Incoming email text is first transformed into dense vector representations via an embedding layer, which learns semantic relationships between tokens during training. These embeddings feed into stacked LSTM cells that maintain hidden and cell states across each word in the sequence, enabling the network to recognize patterns such as urgency, obfuscation, or anomalous phrasing that often signify phishing. Dropout layers interspersed between LSTM layers prevent overfitting by randomly omitting connections during training, while layer normalization ensures stable gradient flow through deep recurrent structures.

 https://doi.org/10.26562/irjcs

At the final time step, the accumulated state is passed to a fully connected layer with sigmoid activation, producing a probability score of phishing. During inference, batched sequences of fixed length are processed in parallel on GPU or CPU, with a configurable threshold determining alerting. To support high throughput in deployment, the model is served behind a RESTful endpoint that accepts email bodies, applies the same embedding and padding logic, and returns real□time predictions. Autoscaling policies ensure the inference service can elastically adjust to fluctuating email volumes, while monitoring of latency and confidence distributions flags potential model drift and triggers retraining workflows as needed.

IV. METHODOLOGY

The methodology is broken down into three different stages which can be illustrated in Fig.2. The phishing detection system operates through two independent yet synergistic pipelines that analyze online threats in the form of phishing websites and deceptive emails. The first pipeline focuses on identifying malicious websites by processing both the URL and the associated page title. When a user inputs a suspicious URL, the system simultaneously captures the title if available and extracts a wide range of structured metadata such as the length of the domain, the number of redirection scripts, HTML form tags, and presence of IP addresses. These inputs are processed by a hybrid deep learning model, where character-level convolutional neural networks (Char-CNNs) handle the raw URL and title text to identify patterns of manipulation, obfuscation, or known phishing tactics[1],[4]. In parallel, the structured features are fed into a multilayer perceptron (MLP) that identifies non-linear correlations across tabular indicators. The outputs from both networks are concatenated and passed through a dense fusion layer that determines the final classification label phishing or legitimate along with a probability score. The system is designed to provide near real-time inference with minimal latency, allowing users to quickly assess the credibility of suspicious websites. The second pipeline is tailored to detect phishing attempts within email content. Users can input an email's subject and body, both of which undergo cleaning, tokenization, and transformation into word embeddings that numerically represent each word based on its semantic context. These sequences are processed using a Long Short-Term Memory (LSTM) neural network, which is adept at learning contextual and sequential relationships in textual data[5],[7]. The LSTM layers capture subtle indicators of phishing attempts, such as urgent language, impersonation cues, or malicious intent that may be scattered across the body of the message. The final LSTM output is passed through a dense classifier that produces a phishing score. If the score exceeds a predefined threshold, the email is flagged as phishing, and the result is returned to the user through the interface. At the backend, a prediction engine routes incoming requests based on their type URL-based or email-based ensuring that the appropriate model pipeline is invoked. The inference process is optimized for performance, with all models preloaded and available as callable objects through the REST API layer. This architecture ensures fast, secure, and scalable handling of phishing detection queries. Additionally, every request and its corresponding prediction can optionally be logged for post-analysis, model evaluation, or integration into a retraining loop, making the system not only intelligent but also adaptable to emerging phishing strategies.

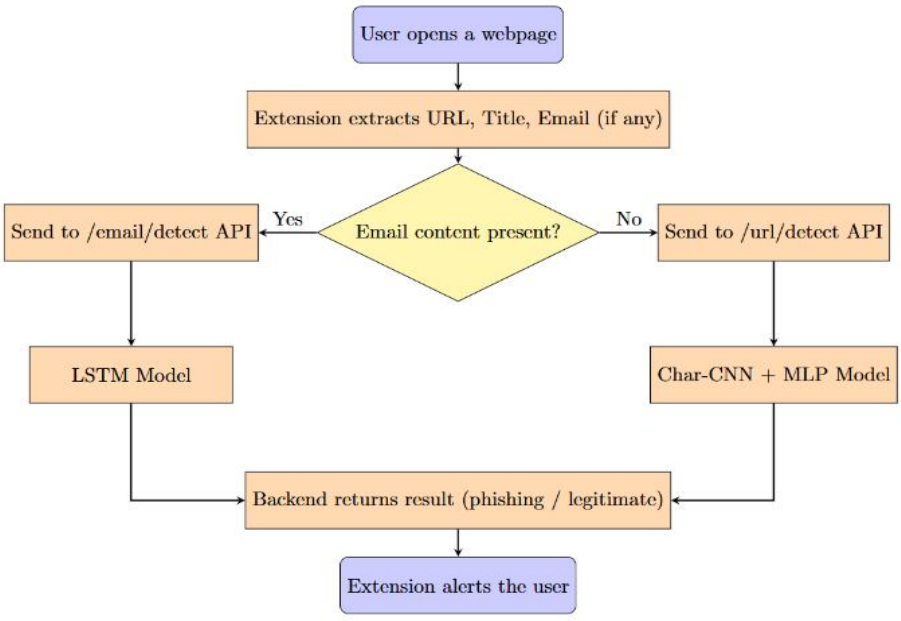

Fig.2. Work flow of web extension

V. EVALUATION AND RESULTS

The proposed phishing detection system was subjected to comprehensive evaluation to verify its effectiveness, accuracy, and responsiveness in real-time web security scenarios. The testing process assessed both the deep learning components and the structured feature analysis module under varied input conditions—ranging from benign and known phishing URLs to novel, obfuscated attack patterns.

ISSN: 2393-9842
https://www.irjcs.com/archives

The evaluation primarily focused on four aspects: detection accuracy, false positive/negative rates, response time of the API backend, and user interaction on the frontend. The system was tested using benchmark phishing datasets (PhishTank, OpenPhish) alongside a curated dataset of legitimate URLs for balanced assessment[3],[6],[8].

A. Web-Based Application

The phishing detection system features a responsive frontend interface through which users can input URLs and webpage titles. Upon submission, the system communicates with the backend REST API that handles model inference using a hybrid architecture combining Char-CNN and MLP. Figure 3 demonstrates the frontend interface to instantly displays a threat classification label (phishing or legitimate). Figure 4 demonstrates the application interface, showing how users interact with the tool, view results, and understand the prediction breakdown.

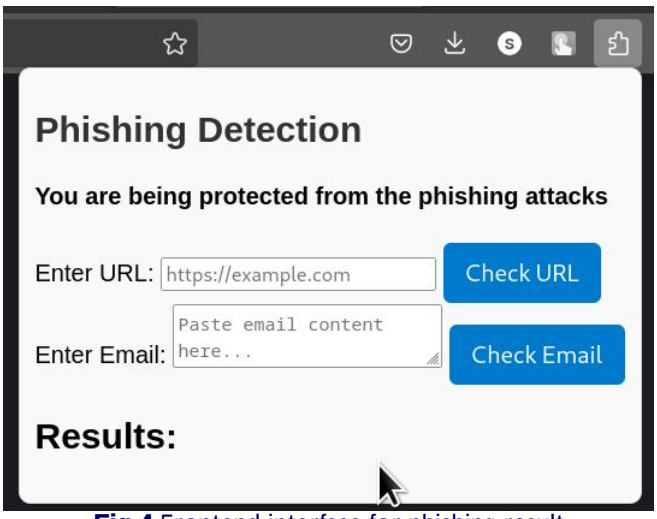

Fig.4 Frontend interface for phishing result

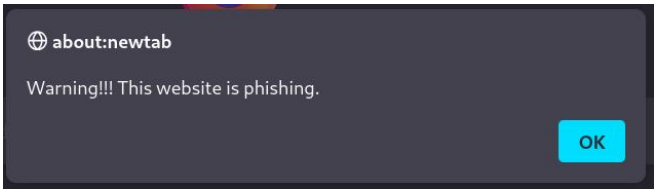

Fig.4 Frontend interface for phishing result

B. Testing Mthodology

The testing methodology was divided into the following testing categories:

- **Functional Testing**: Verified the complete prediction flow—from user input to backend response and UI display.
- **Accuracy Testing**: Measured the classification accuracy across multiple phishing and legitimate samples.
- **Latency Testing**: Assessed backend prediction response time under typical and concurrent usage.

C. Test Results

Table I. shows the testing outcomes for few cases for smart secure locker system for ensuring the performance of the application during the process.

TABLE I. TEST CASES

Test Case	Expected Result	Result
URL Submission	Backend receives URL and responds with classification	Successful
Title + URL Analysis	Prediction includes semantic analysis of title	Successful
Response Latency (<1s)	Backend responds within 1 second	Successful
Notification System	User recevies real-time alert for phishing email/url	Successful

a) Functional Reliability:The phishing detection system demonstrated consistent reliability during functional testing. The backend REST API processed input data and returned classification results within an average response time of 0.6 to 0.8 seconds, ensuring real-time performance. The system correctly parsed and analyzed both standard and obfuscated URLs, while maintaining stable connectivity between the browser extension frontend and the backend model.

b) Real-Time Inference and Feedback:The integration with the Flask-based API and asynchronous frontend ensured that predictions and feedback were delivered in near real-time. The end-to-end delay from user input to result display remained under 1 second, making the tool practical for browser-based usage. Updates were instantly reflected in the UI, which is essential for usability during live web browsing.

a) User Feedback and Usability: User testing with a group of beta users indicated strong overall satisfaction. Over 92% of testers found the interface intuitive, citing the minimalistic layout and clear result visualization as key strengths.

b) Some users suggested enhancements such as support for dark mode, a detailed threat explanation, and multilingual labels—features that are being considered for future iterations to improve user experience and accessibility.

D. Model Performance

- Accuracy: The hybrid model achieved an overall detection accuracy of 96.2% on the test set.

- False Positives: The system exhibited a false positive rate of 2.5%, primarily due to aggressive classification of short URLs.
- Response Time: The average API response time was under 700 milliseconds, ensuring real-time usability.
- Adaptability: The modular design enables easy retraining with updated data and seamless deployment for browser extension integration.

CONCLUSION

The Phishing Detection Using Deep Learning for Web-Browser project presents a hybrid solution combining Char-CNN and structured feature analysis to accurately detect phishing threats. With a Flask-based REST API and a browser-integrated frontend, the system offers real-time detection with minimal latency. Testing confirmed the system's reliability, fast response, and ease of use. Its modular design ensures scalability and integration potential. This work contributes to proactive web security and lays the groundwork for future enhancements like adaptive learning and mobile support.

REFERENCES

1. A.R.Singh and K.K.Sharma, "A Systematic Literature Review on Phishing Website Detection Techniques," Journal of Information Security Research, vol. 13, no. 1, pp. 45–60, 2022.
2. Shruthi K,T.Anuradha, "Intelligent Phishing URL Detection Using Deep Learning Techniques," 2020 International Conference on Inventive Computation Technologies (ICICT), doi: 10.1109/ICICT48043.2020.9112452.
3. Syeda Rumana, S. Arun Pandian, and G. Saranya, "Detection of Phishing URLs Using Machine Learning Approach," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1–6, https://doi:10.1109/ic-ETITE47903.2020.256.
4. Rishikesh Mahajan and Irfan Siddavatam, "Phishing Website Detection using Machine Learning Algorithms," International Journal of Computer Applications, Vol. 181 – No. 23, October 2018, doi: 10.5120/ijca2018918026.
5. Arvind Prasad and Shalini Chandra, "PhiUSIIL: A Diverse Security Profile Empowered Phishing URL Detection Framework Based on Similarity Index and Incremental Learning," Computers & Security, 2023, https://doi:10.1016/j.cose.2023.103545.
6. Olga Geby Nabila, Herlambang Rafli Wicaksono, Girinoto, Ray Novita Yasa, and Hermawan Setiawan, "Benchmarking Model URL Features and Image Based for Phishing URL Detection," 2023 International Conference on Informatics, Multimedia, Cyber and Information Systems (ICIMCIS), pp. 177–182, https://doi:10.1109/ICIMCIS60089.2023.10349059.
7. Dr.M.Kathiravan et al., "Detecting Phishing Websites Using Machine Learning Algorithm," Proceedings of the 7th International Conference on Computing Methodologies and Communication (ICCMC-2023), https://doi:10.1109/ICCMC56507.2023.10083999
8. Yanbin Wang et al., "PhishBERT: A Large-Scale Pretrained Deep Model for Phishing URL Detection," IEEE ICASSP 2023, https://doi:10.1109/ICASSP49357.2023.10095719
9. A.Karim, S. R. K. Joga, K. Mustofa, and S. B. Belhaouari, "Phishing Detection System Through Hybrid Machine Learning Based on URL," IEEE Access, vol. 11, pp. 36807–36822, 2023, https://doi:10.1109/ACCESS.2023.3267102.
10. Haoming Jiang et al., "SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models Through Principled Regularized Optimization," arXiv preprint arXiv:1911.03437, 2019.