iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79315

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Automated Machine Learning Approaches for Phishing Detection and Prevention

Dr. M V D S Krishna Murty[1], Mirupathi Sathvik Reddy[2], Rohit Kumar Sharma[3], Mohammed Nayyar[4]
[1]*Associate Professor, Department of Computer Engineering, Methodist, College of Engineering and Technology, Abids, Hyderabad, Telangana, 500001, India*
[2, 3, 4]*Student, Department of Computer Engineering, Methodist College of Engineering and Technology, Abids, Hyderabad, Telangana, 500001, India*

Abstract: ***The most common type of financial cybercrimes in the highly digitalized economy of India is also identified to be social engineering attacks/phishing. According to the Ministry of Home Affairs, in the financial year 2023-24, cybercriminals have withdrawn ₹22,845.73 Crores from citizens through online scams. It is also identified in the context of global cybercrimes that in the year 2023, the number of attacks recorded by the Anti-Phishing Working Group is 4.9 million. This is the highest number of cybercrimes ever recorded in history.The tools used for detection are mostly unimodal and reactive, utilizing 'black box' systems that are impossible to interpret. This project proposes PhishShield, which is a proactive multi-vector AI framework used for detecting cyber threats in real time, which is developed in the form of a publicly available Web Dashboard as well as a lightweight Chrome Browser Extension. This AI framework utilizes two highly efficient machine learning classifiers, namely, Naive Bayes Classifier with TF-IDF Vectorization for semantic markers in unstructured text data such as SMS and Email, as well as Decision Tree Classifier for structural features in URL data. The accuracy of the URL classifier is enhanced with the integration of Real-Time OSINT Heuristics. The output of the AI framework is analyzed, and the results are compared with the output of the Explainable AI module, which provides a definite two-class output in the form of 'Verified Safe' and 'Threat Detected' with Logic Trace. The testing of the AI framework is conducted, which provides promising results in the form of 92.6% accuracy with Naive Bayes NLP Classifier for detecting semantic threats as well as 90.7% accuracy with Decision Tree Classifier and OSINT Heuristics for detecting deceptive URLs.***

Keywords: Phishing detection, Explainable AI, Real-Time OSINT heuristics, Browser security, Social engineering detection, URL metadata.

I. INTRODUCTION

A. Background Context

In the last ten years, an exponential growth rate has been observed in the digital world of India, with digital technologies such as UPI and Aadhar reaching 950 million active internet users in 2024. With the emergence of digital technologies, an exponential growth rate in sophisticated cybercrimes has been observed. The Ministry of Home Affairs has stated that the financial loss due to cybercrimes has crossed an astonishing figure of ₹22,845 crore for the year 2023-24. Moreover, this figure is three times higher compared to the previous years. At the same time, the Global Anti-Phishing Working Group has stated that nearly 5 million unique phishing attacks have been observed in the year 2023, which is the highest ever. The rising menace of cybercrimes is targeting the newly added digital population through sophisticated social engineering techniques to bypass traditional security mechanisms.

B. Global and Indian Context of Phishing

Though it is agreed that the overall scope of phishing activity is worldwide in nature, it is also understood that it is having a very disproportionate impact on different regions of the world. In order to understand the scope of this problem at the worldwide level, as well as the overall scope of this problem for India, it is necessary to understand the overall design elements that have been incorporated within PhishShield.

In the year 2023, it was seen that there was an increase in phishing activity at the worldwide level, with an increase of 76% in mobile-based phishing attacks and more than 1.3 million unique malicious websites identified in the fourth quarter of 2023. While it is understood that the West is facing corporate-centric phishing attacks, it is India that is facing the maximum number of mobile-based phishing attacks due to the usage of smartphones and SMS services.

©IJRASET: All Rights are Reserved

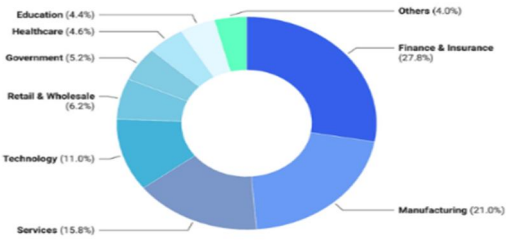


Fig 1: *Global Distribution of Phishing Impact and India's Exposure Profile*

Presently, India ranks second in the world in terms of phishing victimization. In this case, India has a high rate of financial cybercrimes through 67% of SMS-based frauds. However, this can be attributed to a large gap in terms of the rate of digitization and digital illiteracy in India. For instance, 85% of rural areas in India are not able to identify the symptoms of phishing.

C. Linguistic and Socio-Technical Challenges in India

Although phishing has proved to be an omnipresent threat, the consequences of this threat, however, seem to be dis-tributed unevenly around the world. The aim of this section, therefore, is to develop an understanding of this particular phenomenon from a global perspective, moving towards a local perspective, so that a better understanding can be developed regarding how this particular decision was made to develop PhishShield. For example, statistics reported across the globe in the year 2023 have reported an unprecedented increase in terms of threat volumes, where it was reported that there was a 76% increase in terms of mobile-centric threats, along with more than 1.3 million unique malicious do-mains detected in Q4 alone. While phishing attacks originating from the West target corporate infrastructures, India has turned out to be a hotspot for mobile-based phishing attacks, primarily due to its large population and its culture of using SMS and other digital tools of communication.

D. Motivation for PhishShield

PhishShield is a highly innovative multi-vector detection system that focuses on addressing the significant operational weaknesses of the current state of art in web browser security. PhishShield not only focuses on the evaluation of "link" but also focuses on the evaluation of "structural URL metadata" and "unstructured semantic content" of social engineering-based threats. PhishShield utilizes the "power of parallel machine learning algorithms." PhishShield utilizes a Naive Bayes classifier along with TF-IDF for text-based evaluations, as well as an "optimized decision tree" along with multivariate evaluations of URLs. PhishShield is highly efficient in achieving "high accuracy" in evaluating "threats in the complete absence of user interaction." PhishShield also focuses on addressing the "Reactive Gap" utilizing the "power of OSINT-based heuristic evaluations." PhishShield utilizes "WHOIS-based domain age evaluations" for recognizing the "high volatility and dynamics of threats." PhishShield utilizes "an alternative approach to traditional 'black box' evaluations" utilizing the "Explainable AI (XAI)" module, where "complex evaluations of threats are translated into educational 'Logic Traces' for vulnerable users".

E. Objectives of the Project

The main goal here is to develop a framework called PhishShield, which is a multi-vector framework to tackle multi-modal attacks through integrating unstructured text evaluation using the Naive Bayes and TF-IDF NLP pipeline for detecting social engineering attacks such as financial urgency and code-mixed phrases. This framework also aims to integrate structured data evaluation using Decision Tree and OSINT heuristics such as WHOIS and SSL for detecting zero-day URLs. This framework aims to be developed as a Manifest V3 Chrome Extension and Web Dashboard with a dynamic input router using Flask and Explainable AI for integrating machine learning using Explainable AI for "transparent Logic Traces." This framework also aims to integrate a closed-loop human-in-the-loop framework with integrated adversarial retraining using SQLite database.

II. LITERATURE SURVEY

In this section, an overview of the current literature on phishing and social engineering detection is given, with regard to rule-based methodologies, traditional machine learning techniques, and structural metadata analysis. The deployment problems, the possibility to process in real time, and the need to provide explainable security are important in the development of PhishShield.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

A. Rule Based and Traditional Approaches

Traditional lightweight security applications may employ reactive blocklists based on URLs and scoring matrices. Re-search has repeatedly confirmed that reactive security systems have detection delays of several hours. In fact, zero-day malicious domains can go undetected for up to 21 hours. These static systems simply cannot cope with the dynamic nature of modern phishing infrastructures. These systems face particular challenges when it comes to dealing with code-mixed languages. These limitations of reactive blocklists and keyword-based detection systems highlight the im-portance of the proactively employed machine learn-ing based text analysis and real-time OSINT verification process employed by PhishShield.

B. Classical Machine Learning Approaches

This is regarded as a major achievement in terms of utilizing machine learning technology, where classical machine learning techniques have been capable of identifying probabilistic patterns associated with threats through a set of data.Classical machine learning techniques like SVM, Random Forest, Naive Bayes, etc., have been considered promising for identifying phishing attacks. It has been observed that Naive Bayes Classifier, especially when utilized in combination with optimal feature extraction techniques, promises to deliver promising results for identifying semantic social engineering indicators with low computational complexities. In addition, Decision Trees have been considered promising for analyzing structured multivariate data, including URL metadata and lexical anomalies. Even though it has been identified in recent research that classical machine learning techniques cannot be considered suitable for this problem, considering detailed feature engineering techniques, this feature has been considered major for the PhishShield system.

C. Deep Learning Architectures

In recent times, academic studies have focused on the application of deep learning models such as Convolutional Neural Networks and Recurrent Neural Networks due to the ability of these models to learn features automatically from the data and achieve detection rates above 98%.

However, despite the high detection rates obtained with deep learning models, these models are highly challenging in terms of operations for client-side deployment. The models require high computational power, including the need to support NVIDIA GPU cards and memory requirements, making them unsuitable for deployment in lightweight clients such as Chrome browser extensions. Moreover, deep neural networks are essentially "black boxes" that fail to offer the transparency needed to offer an educational, user-friendly experience for security feedback. Thus, PhishShield avoids using computationally complex architectures, instead using highly optimized classical machine learning models with live OSINT heuristics.

D. Transformer and Large Language Model

In the recent NLP studies carried out on the application of NLP in phishing detection, Transformer architectures such as BERT have been emphasized for the detection process due to the presence of the bi-directional attention mechanism that provides deep contextual embeddings and helps in the detection of the subtle tactics of social engineering with high F1 scores.

However, it is seen that even the light versions of the BERT architecture, such as DistilBERT, incur huge latency and computational costs and are not appropriate for the implementation in the browser. Also, the complex and black-box nature of the Transformer architecture contradicts the Explainable AI (XAI) requirement. Therefore, PhishShield avoids the implementation of the Transformer architecture and instead uses an optimized Naive Bayes classifier with TF-IDF vectorization. The classical approach provides the advantage of processing the data in less than one second with the ability to trace the logic.

E. Multi-Vector and Hybrid Detection Systems

The phishing attacks of the modern era are able to evade the single vector system with ease by concealing the zero-day malicious links with extremely convincing socially engineered texts. Although the integration of linguistic and structural analysis is required to detect the compounded phishing attacks, the integration of multiple data sources faces the problem of considerable latency.

To overcome the problem of latency in the detection of phishing attacks, the PhishShield tool makes use of the multi-vector fusion framework with optimized parallel processing. The tool bypasses the computationally expensive process of OCR and directs the input in such a way that the text is passed to the Naive Bayes NLP engine for semantic evaluation and the URL is passed to the live OSINT-augmented Decision Tree for structural verification.

F. Advanced and Adversarial Techniques

As phishing attacks are becoming increasingly sophisticated, it is important that detection strategies also advance and keep pace with adversarial attacks like data drift and semantic obfuscation. While there are many strategies, as proposed in the latest literature, that heavily rely on computationally expensive techniques like Generative Adversarial Networks for generating adversarial phishing at-tacks for model hardening, such kinds of attacks are often not realistic.

Also, whereas complex deep learning models such as Temporal Convolutional Networks (TCN), in an attempt to make these models post-hoc explainable and better comprehend the relative importance of features in the data provided to the model, are only able to yield probabilistic and non-explanatory results that are hard to comprehend for the average layperson, the PhishShield Explainable AI module enables the achievement of highly transparent and deterministic results. This is largely because, by explicitly converting Naive Bayes TF-IDF keyword weights and live OSINT intelligence such as "Domain Age < 24 hours," "Invalid SSL" into highly comprehensible "Logic Traces," the PhishShield framework enables the achievement of a highly adaptive and operationally viable solution that is always able to effectively counter and adapt to highly dynamic and hostile security environments.

G. Operational and Deployable Systems

Besides performance problems, there are significant challenges in terms of applicability in terms of user interface and system flexibility. The literature suggests that system modularity is an important requirement for system resilience. This is due to its ability to update system components concurrently without interfering with other components. The flexibility of PhishShield is attributed to its backend technology, Flask, which enables concurrent optimization of its Naive Bayes NLP and Decision Tree URL classifier. PhishShield is also easy to deploy as a Chrome Extension/Web Dashboard for immediate verification at points of interaction.

One of the main reasons for this fatigue and mistrust is the presence of deciphering false positives which have been incorrectly classified by an unintelligible algorithm. As users are repeatedly denied access without explanation, they simply cease to pay attention to any security messages. In order to alleviate this problem, PhishShield simply ignores these unintelligible messages in favor of an unequivocal determination of Verified Safe or Threat Detected, without any input from any other portion of the system except for an Explainable AI (XAI) module. By giving users an intelligible "Logic Trace" in the form of identifying a new domain via WHOIS or invalid SSL certificate, the system is working to alleviate fatigue due to changing data.

H. Dataset and Evaluation Issues

Moreover, it is also observed that the quality of training data is the major factor in deciding the effectiveness of phishing detection frameworks. The majority of the studies are based on outdated data sets such as the UCI SMS Spam Collection Data Set, which is not capable of generalizing against modern social engineering attacks and zero-day URLs. To achieve the effectiveness of Phish-Shield in real-world scenarios, the NLP classifier and multivariate decision trees are trained on diverse data sets. Moreover, significant differences in the evaluation of the proposed frameworks are also observed in the literature. The majority of the studies have in-consistently focused on static evaluation criteria rather than deployability. An algorithm might be highly accurate in terms of F1-Score in offline scenarios but is of no use if it results in significant latency in real-world scenarios. To overcome this limitation of the proposed frameworks, PhishShield extends the overall evaluation criteria of the proposed frameworks. PhishShield is not only tested in terms of the overall accuracy of machine learning pipelines in classifying URLs as malicious or benign but is also subjected to benchmarking in terms of overall latency in sub-second timescales and overall clarity in Explainable AI logic.

I. Legal and Social Dimensions

However, it is also important to note that the detection of phishing is not just related to the technical aspects of the algorithms. Rather, it is also linked with the legal and social frameworks. For instance, in the context of the digital economies of India, it has been seen that the cyber laws do not address the exponential growth in financial frauds, particularly in the context of smishing and UPI. However, it is important to note that although the regulatory bodies such as the Information Technology Act of 2000 and the Telecommunication Regula-tion of India’s Distributed Ledger Technology system are focused on controlling unsolicited and malicious communications, the gap in the regulatory enforcement is creating risks for the end-users in the context of sophisticated social engineering attacks.

From a sociological point of view, traditional digital literacy and awareness strategies seek to educate these vulnerable populations, especially in rural or newly connected demographics. However, such strategies are static in nature, provide low retention, and, most importantly, are not scalable to meet the demands of the current zero-day attacks. Therefore, the need for automated and proactive security measures has never been more important.

PhishShield addresses this significant sociotechnical gap by going beyond static and background measures of security. Using the Explainable AI (XAI) component, "Logic Traces" are created, allowing for the proactive education of the user, thus promoting long-term cyber resilience and autonomous recognition of threats.

J. Summary and Research Gaps

Although it is a fact that with the evolution of phishing detection from simplistic filters to sophisticated and complex neural networks, there is an increase in the accuracy of the methods, it is also a fact that there are shortcomings in the operation of the methods. It is a fact that the methods that are in place today are computationally expensive, singular in their approach, and divide the text and the network in order to perform the detection, but it is also significant that the methods in place today are only geared towards the detection and prevention of phishing attacks, whereas they are not geared towards preventing them in the first place. Perhaps the most significant short-coming of sophisticated and complex neural networks is the fact that, in their inability to explain the reasoning behind their decisions, they leave the average user, most especially those who are not as computer savvy, in complete darkness.

III. SYSTEM ARCHITECTURE

A. Architecture Overview

The architecture of PhishShield has been specially designed in such a manner that it can provide an effective, multi-vector, and highly responsive defense system in response to the current threats of social engineering and phishing attacks. In light of the complexities of current threats, which may include malicious infrastructures in addition to false text messages, a multi-layered architecture has been developed through an effective breakdown of input data and then integration with the help of an integrated system of risk evaluation. This is effective in ensuring that all parts of the system are working effectively in such a manner that it considers the latency constraints of real-time environments.

The architecture is specifically designed in such a manner that there are four specific phases. Each of these phases is specifically designed for handling a specific part of the entire real-time detection mechanism. The first phase is specifically designed for handling the collection of data on the client-side, specifically through the Chrome Extension or the Web Dashboard. The second phase is specifically designed for handling the dynamic splitting of data, specifically in terms of unstructured text data as well as structural URL data. Both of these are then subjected to highly optimized machine learning pipelines, specifically in terms of Naive Bayes for semantic text data as well as Decision Trees with live OSINT heuristics for URL metadata. The result of both of these is then used for arriving at a definitive classification. This is then followed by the educational XAI. The entire purpose for which this mechanism is designed is specifically for the system to be able to perform phishing detection in an efficient manner. This is absolutely necessary in real-world scenarios.

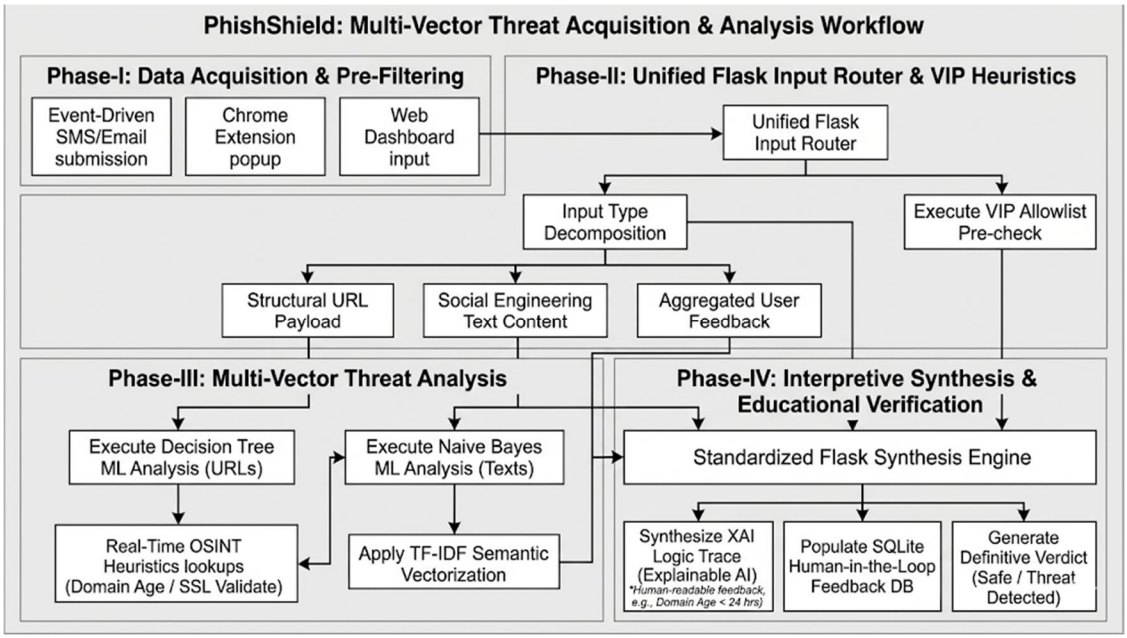


Fig 2: System Architecture for PhishShield

B. Detailed Explanation of the Four Phases

1) Proactive Client Side Acquisition

The threat detection pipeline begins when the user encounters the suspicious content and submits it to the PhishShield system through the PhishShield Web Dashboard or the Chrome Browser Extension popup. The acquisition process is event-driven, which makes it easier to optimize the computational overhead of the system. After the user submits the suspicious content, the standardized Flask Input Router receives the data and immediately feeds back the interface to the user. The process is the entry point of the entire verifica-tion process.

2) Unified Input Decomposition & Heuristics

At this stage, the centralized flask-based router would automatically disassemble the payload into separate operational vectors such as Social Engineering Text Content and Structural URL Payloads. Before the computationally intensive machine learning process is called, an immediate VIP Allow List Pre-check is performed. It is imperative to note that this filtering process is important in that it ensures that only the domains that have been validated are immediately cleared in the most time-efficient way possible.

3) Multi-Vector Threat Analysis

Subsequent to this, various data types are passed on to their respective parallel classification pipelines. For example, unstructured text data is passed on to the Natural Processing engine, where TF-IDF semantic vectorization takes place, after which the data is passed on to a finely tuned Naive Bayes classifier, which detects linguistic manipulations and urgency signals. At the same time, URLs are passed on to an optimized Decision Tree model. Structural analysis has been strictly augmented with real-time OSINT heuristics, where instantaneous live lookups for domain age volatility and SSL certificate validation take place, aiming to detect zero-day infrastructure.

4) Interpretive Synthesis & Educational Verification

Lastly, in the final phase, the Standardized Flask Synthesis Engine combines the results that have been obtained in the NLP and OSINT-based URL parallel pipeline. The user is now provided with a definite verdict in the form of ‘Verified Safe’ and ‘Threat Detected,’ instead of probabilities that are ambiguous and uncertain in nature. However, the most important feature is that the classification is now provided automatically with the Explain-able AI Logic Trace. The user is also provided with the integrated SQLite Human-in-the-Loop database.

IV. IMPLEMENTATION

A. Development Environment and Technology Stack

There are three major phases of development in PhishShield, which include model training, backend development, and frontend development. In the training of the model, the development environment is Python, which uses the Scikit-learn library to develop the Naive Bayes Classifier and the Decision Tree Classifier. The CPU is used exclusively for the training of the model to meet the lightweight deployment criteria. In the backend development, the routing API is developed using Flask, and the OSINT libraries, such as Python Whois and Ssl, are used in real-time network heuristics. Dependencies are rigorously handled during this phase of development. In the frontend development, the Chrome Extension is developed using standard web technologies with strict Manifest V3 compliance. Phase I is where user interaction is used to initiate the start of the threat verification process in the PhishShield framework. Unlike in other highly intrusive background scanning processes, where the system is continuously monitoring all local messages, an event-driven design is a special aspect of the PhishShield framework, which is highly user-centric and also resource-friendly on the host device.

1) Pipeline 1 Implementation:Semantic Text Classification

The first pipeline is based on data pre-processing, vectorizing models, and real-time classification. In the data pre-processing stage, open-source phishing messages are collected in standard format. This is done by converting all formats to a rigid standard, which means threat is equal to 1 and safe is equal to 0. In addition, data cleaning is performed by eliminating duplicate messages and introducing noise removal. In this case, text data is vectorized in accordance with TF-IDF vectorization.

For the training of the model, the Multinomial Naive Bayes Classifier is used due to its high speed performance. Also, it is compatible with the TF-IDF matrix. The model is saved based on its performance over the validation set, which is robust with 92.6% accuracy. In the live deployment scenario, it is capable of loading the light model instantly. Also, it is very efficient in processing the messages without going into complex calculations of deep learning gradient models, which results in the threat probability score in milliseconds.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

2) Pipeline 2 Implementation:Metadata Feature Extraction and OSINT

As far as the implementation of the structural URL processing technique is concerned, it can be divided into two parts, namely the dynamic feature extractor utility class, which is responsible for processing the lexical information in the URLs, and the decision tree classifier, which is very intimately coupled with the live OSINT network heuristics.

In the feature extractor class, feature engineering is performed through aggressive parsing of the attributes in the URLs, which can be done through the use of standard libraries for the accurate parsing of the URLs. The entropy of the characters in the URLs is also calculated in order to determine patterns that are created through algorithmic processes as well as those created through URL shorteners. Most importantly, it performs real-time WHOIS lookups for the age of the domains as well as the validation of the SSL certificates.

3) Fusion Module and XAI Implementation

This Synthesis module is where all of the individual results from the various text and metadata processing blocks are combined, and the final classification of the risk is determined. It is also "intelligent" in the fact that it can adapt in accordance with the non-existence of certain types of information, i.e., if only the URL is provided to the system and no text, then Na-ive Bayes is completely bypassed, and the system relies only on the results of the Decision Tree and the OSINT heuristics. It strictly classifies the input as Verified Safe or Threat Detected.

It also has an Explainable AI (XAI) feature that provides complete transparency, enabling the viewing of the "Logic Trace" of the classification process. Moreover, in the case of text payload classified as high-risk, it will highlight suspicious semantic keywords, and in the case of metadata, it will highlight risky attributes such as the presence of an obfuscated URL shortener and the domain name being registered in less than 24 hours.

4) Deployment and System Integration

The process of deployment of the PhishShield product is a highly automated and streamlined process of continuous integration, which directly integrates the GitHub code repository with the live environments with minimal human interaction. The client-side interface of the Chrome Extension includes a highly automated process of deployment using the official Chrome Web Store Developer Dashboard for hosting, verification, and publication. Further, a highly strict process of GitHub Actions is included for every code merge into the main branch of the code repository, along with scripts being implemented for a highly strict Manifest V3 compliance and the packaging of the Chrome Extension code for the end-user update process. Perhaps the most important part of the entire process is the fact that the configurations are set up as environment variables and not on the frontend.

The routing structure is based on a scalable cloud virtual machine running an application with Flask. During initialization, it loads highly optimized Naive Bayes and Decision Tree models into memory and starts the application with Waitress to process multiple client verification requests in parallel. Nginx is used as a reverse proxy with security through enforcement of the HTTPS protocol and implementation of rate limiting and comprehensive system logging. Retraining of defense mechanisms is simply a case of replacing .pkl files in the backend folder and reloading the application without changing any of the extension code, which is due to the modularity of the parallel processing approach.

V. METHODOLOGY

The underlying motivation for developing the architectural framework for PhishShield is to create an effective defense system that questions the extremely deceptive nature of social engineering. While other conventional security filters only verify for text strings that contain known malicious keywords, our framework recognizes an underlying reality that states that a true security threat will emerge in an amalgamation of semantic ur-gency and underlying structural anomalies in an embedded zero-day URL.

A. Multi-Vector Architecture

We have specifically designed our PhishShield system with the ability to intercept sophisticated phishing attempts using concurrent communications analysis on two distinct operational vectors. In order to effectively counter deceptive conversational schemes, we have implemented a highly optimized Naive Bayes NLP system that incorporates TF-IDF vectorization. This highly sophisticated mathematical system is capable of identifying underlying semantic intentions and financial urgency in any given message, making it impossible for an attacker to bypass this system using text obfuscation techniques. Furthermore, our system’s second vector bypasses computationally complex and often inaccurate image processing altogether and relies exclusively on structural integrity.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

This is done through the employment of a specific classifier, termed the Decision Tree classifier, which strictly analyzes incoming links through thousands of patterns by employing metadata within the network. This allows our system to spot nuanced lexical anomalies, routing, and structural red flags in an instant, making it an effective countermeasure for both the communicative and technological components of a specific threat.

B. Live OSINT and XAI Synthesis

However, because of the ineffectiveness of static analysis in determining changes in infrastructure, we have chosen to forego the reliance on pre-trained model weights. Moreover, we have expanded our structural analysis to include Real-Time OSINT heuristics to validate domain age and SSL validity in real-time. Rather than using vague voting systems, our Standardized Flask Synthesis Engine utilizes a deterministic aggregation of the NLP/OSINT data. By concurrently examining two forms of data, our program effectively emulates the way an educated security researcher would manually investigate a potentially malicious payload—intensely examining what it is attempting to coerce the user into doing, but at the same time examining the hidden network architecture to arrive at a transparent conclusion.

C. System Realization & Performance

Also, PhishShield is specifically designed for a very tight optimization, considering that it uses a Flask backend with a Chrome Extension that can return the results within less than 500 milliseconds, without any lag on the browser. Moreover, extensive validation tests have proven that our semantic NLP engine can attain a high accuracy of 92.6%, whereas our metadata classifier using OSINT can attain a high accuracy of 90.7%. Moreover, our system does not only display the warning flags but also gives a definite verdict of 'two-class,' along with 'Logic Traces' from Explainable AI to educate users on how to identify the flags.

VI. RESULTS AND DISCUSSION

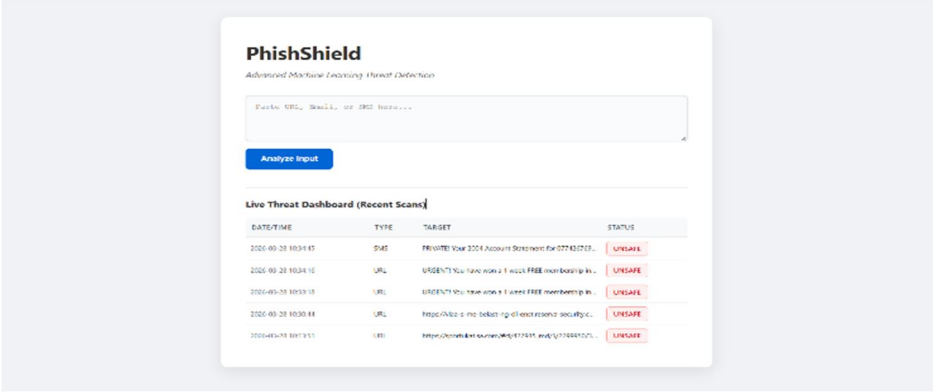


Fig 3: PhishShield Web Dashboard Landing Page

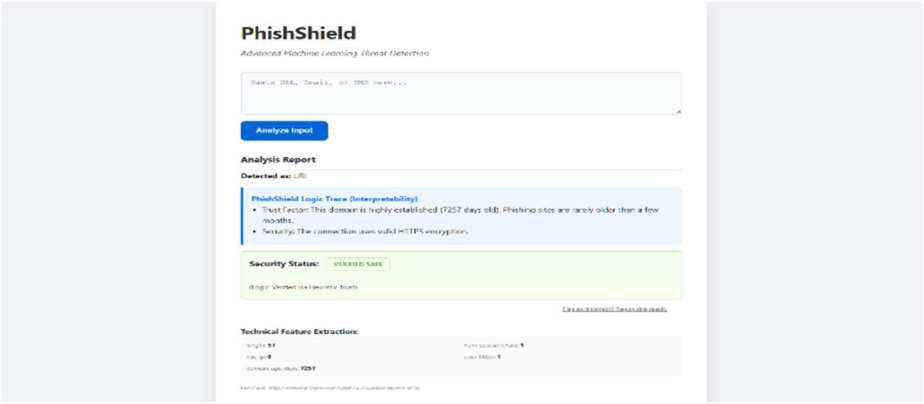


Fig 4: Analysis Report for a Verified Safe URL (Legitimate Domain)

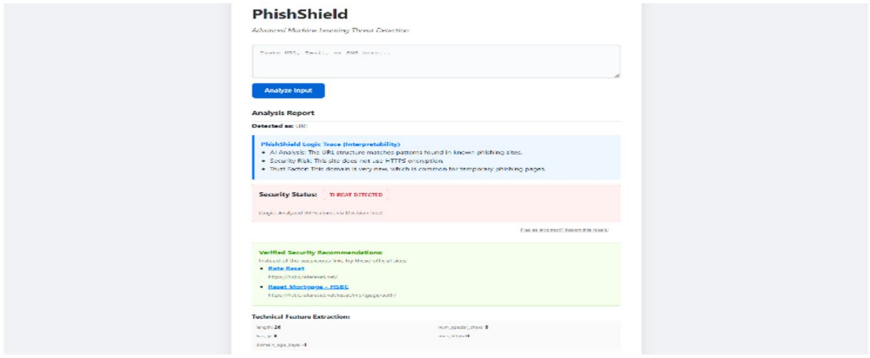


Fig 5: Analysis Report for a Detected Threat URL with Logic Trace and Recommendations

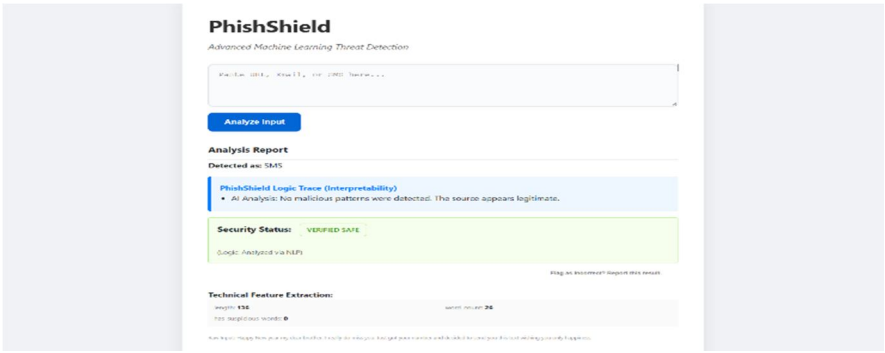


Fig 6: Analysis Report for an NLP-Verified Safe SMS Payload

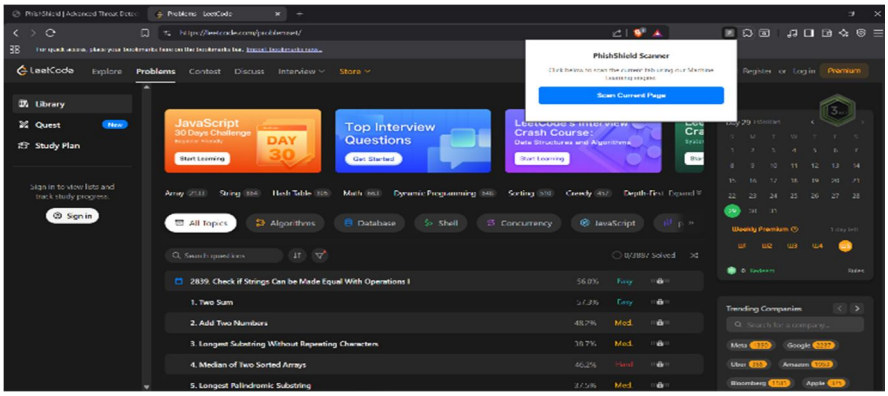


Fig 7: PhishShield Chrome Extension on a Safe Website

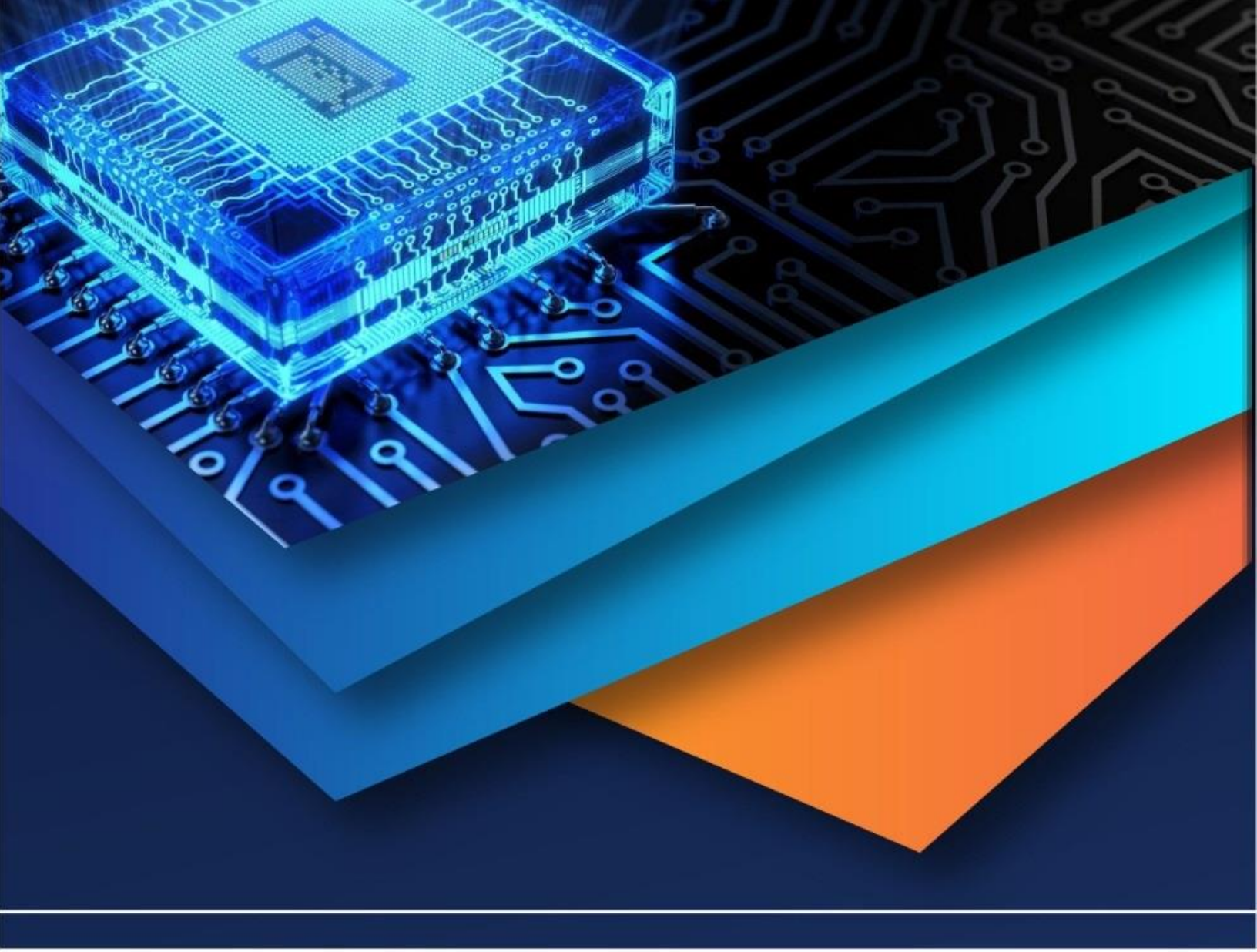
VII. CONCLUSION

This paper proposed the PhishShield framework for the detection of multi-vector social engineering and financial fraud attacks against internet users by integrating semantic textual analysis and structural URL analysis. The deployed framework addresses some of the loopholes in the current reactive defense mechanisms against such attacks, particularly in the context of the black box models that are becoming more prevalent because of the rise in the volume of zero-day malicious infrastructure. The optimization considerations that were taken into account in order to adhere to the sub-second latency requirement in the deployed framework, which is currently deployed in the context of the Chrome Extension, are the following: all of these considerations relate to the refinement of the Natural Language Processing module that utilizes the Naive Bayes classifier and the real-time implementation of the Decision Tree classifier in the context of the live OSINT heuristics, in addition to the computational efficiency of the system in the context of standard browser environments. The area of focus for the future of this line of research would be to extend the logic for Explainable AI (XAI), as well as the SQLite human-in-the-loop adversarial feedback feature. With these objectives set for the future, Phish-Shield is still a very efficient security system for vulnerable users around the world.

REFERENCES

[1] H. S. Lamsal and T. Kumar, "Smishing detection using machine learning algorithms," Journal of Cybersecurity and Information Management, vol. 14, no. 1, pp. 8–19, 2025.
[2] S. S. Banu and P. M. Kumar, "Smishing detection using machine learning algorithms," Tuijin Jishu/Journal of Propulsion Technology, vol. 45, no. 2, pp. 3890–3896, 2024.
[3] R. K. Jha, S. K. Singh, and S. V. N. S. R. Rao, "Smishing detection using machine learning and deep learning," in 2024 2nd International Conference on Disruptive Technologies (ICDT), Greater Noida, India, 2024, pp. 303–307.
[4] D. A. Oyeyemi and A. K. Ojo, "SMS spam detection and classification to combat abuse in telephone networks using natural language processing," Journal of Advances in Mathematics and Computer Science, vol. 38, no. 10, pp. 144–156, 2023
[5] P. S. Rayalla, G. Katakam, S. V. Vivek, H. Golla, and M. Zabeeulla A. N., "Detecting phishing websites using deep learning," in 1st International Conference on Recent Innovations in Computer Science and Technology, 2023.
[6] S. R. A. Samad, P. Ganesan, J. Rajasekaran, M. Radhakrishnan, and H. Ammaippan, "SmishGuard: Leveraging machine learning and natural language processing for smishing detection," International Journal of Advanced Computer Science and Applications, vol. 14, no. 11, pp. 659–666, 2023.
[7] W. L. T. T. N. Kumarasiri, M. K. J. C. Siriwardhana, S. A. D. S. L. Suraweera, A. N. Senarathne, and S. M. B. Harshanath, "CyberSmish: A proactive approach for smishing detection and prevention using machine learning," in 2023 7th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud), Kirtipur, Nepal, 2023, pp. 210–217.
[8] M. Sánchez-Paniagua, E. Fidalgo Fernández, E. Alegre, W. Al-Nabki, and V. González-Castro, "Phishing URL detection: A real-case scenario through login URLs," IEEE Access, vol. 10, pp. 42949–42960, 2022.
[9] N. Noah, A. Tayachew, S. Ryan, and S. Das, "PhisherCop: Developing an NLP-based automated tool for phishing detection," SSRN Electronic Journal, 2022.
[10] T. Wood, V. Basto-Fernandes, E. Boiten, and I. Yevseyeva, "Systematic literature review: Anti-phishing defences and their application to before-the-click phishing email detection," IEEE Access, vol. 10, pp. 1–21, 2022.
[11] O. Abayomi-Alli, S. Misra, and A. Abayomi-Alli, "A deep learning method for automatic SMS spam classification: Performance of learning algorithms on the indigenous dataset," Concurrency and Computation: Practice and Experience, vol. 34, no. 17, Art. no. e6989, 2022.
[12] O. N. Akande, O. Gbenle, O. C. Abikoye, R. G. Jimoh, H. B. Akande, A. O. Balogun, and A. Fatokun, "SMSPROTECT: An automatic smishing detection mobile application," ICT Express, vol. 9, no. 2, pp. 168–176, 2022.
[13] K.-R. Kont, "Cyber literacy skills of Estonians: Activities and policies for encouraging knowledge-based cyber security attitudes," Information & Media, vol. 96, pp. 80–94, 2023.
[14] M. K. Mahadi et al., "A phishing detection approach for empowering cybersecurity with explainable AI," in 2024 27th International Conference on Computer and Information Technology (ICCIT), 2024, pp. 1–6.
[15] A. Verma and S. K. Sharma, "A hybrid machine learning approach for phishing detection using URL and content features," International Journal of Information Security and Privacy, vol. 18, no. 1, pp. 45–58, 2024.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)