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.2026.79410

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

PhishSight AI: Explainable NLP-Based Phishing and Scam Detection with Visual Attention Assist

S. Jayapradha[1], A. Afiga Begum[2], S. Shahika[3], A. Noorul Hasna[4], R. Sowmiya[5]
Department of Artificial Intelligence and Data Science, M.I.E.T. Engineering College, Trichy, Tamil Nadu, India
S. Jayapradha, M.E., Assistant Professor (Project guide)

Abstract: ***Online phishing has quietly become one of the hardest problems in everyday cybersecurity—not because it is technically complex, but because it exploits the one component no patch can fix: human attention. PhishSight AI is designed to assist users where it actually matters, right inside the browser at the exact second a user is about to make a mistake. The system runs as a Chrome extension and checks emails, SMS content, and live web pages against a classifier that combines a Random Forest model, TF-IDF text features, and a fine-tuned BERT network. Every prediction comes with an explanation—LIME and SHAP identify the specific words and URL parts that most affected the output, highlighted directly on screen. An optional gaze-tracking layer using WebGazer.js watches whether users actually look at the sender address and URL bar before clicking; if they do not, a soft prompt reminds them to check. On a held-out test set, the system reached 95.2% accuracy with end-to-end response time under two seconds, demonstrating that real-time detection and explanation can be achieved without noticeable latency.***
Keywords: Phishing detection, Explainable AI, Chrome extension, LIME, SHAP, BERT, Random Forest, URL Feature analysis, Gaze tracking, Cybersecurity.

I. INTRODUCTION

There is a version of this problem that looks simple on paper. Someone sends you a fake email pretending to be your bank. You click a link, enter your password, lose your account. The fix, in theory, is to not click. But that framing ignores almost everything that actually matters about how people use email in practice. Workers process dozens of messages a day. Context switching is constant. Phishing emails have gotten much better—some are now grammatically indistinguishable from legitimate correspondence, built using large language models and designed to arrive at moments of low attention. The Anti-Phishing Working Group logged over 4.7 million incidents in 2023, and the numbers have not been improving.

Machine learning classifiers have gotten good at flagging suspicious content. Several recent systems report accuracy above 96% on benchmark datasets, and that is genuinely encouraging progress. But there is a problem the benchmark numbers do not capture: users do not trust tools they cannot understand. When a classifier blocks a message and says nothing more than "phishing detected," users learn nothing. Our view is that detection and explanation need to be treated as the same problem. A system that tells you which words looked suspicious is doing something qualitatively different from one that only renders a verdict.

PhishSight AI was built around that idea. Four modules handle the actual work: one analyzes email and SMS text, one scans web pages and URLs, one manages the explanation and gaze-tracking interface, and one runs the FastAPI backend and feedback database. The user sees a single popup; the complexity underneath is intentionally hidden. The whole pipeline runs fast enough—under two seconds—that most users do not notice a delay at all.

II. RELATED WORK

Work on automated phishing detection goes back at least to the early 2000s, when rule-based filters and domain blacklists were the dominant approach. One persistent tension is the trade-off between accuracy and interpretability. Hernandes [1] addressed this in 2021, applying LIME and Explainable Boosting Machines to URL feature sets and showing that the explanations were practically useful to analysts reviewing flagged content—framing explainability not as a reporting requirement but as a tool for building user trust.

Pavani et al. [2] pushed classification accuracy higher in 2024, reaching 98.6% with Random Forest on URL features. Alam [3] took an ensemble approach in E2Phish, stacking classifiers with SHAP and LIME explanations for 97% accuracy. Yakandawala et al. [4] trained an SVM on email content paired with LIME, reaching 96%. Gao et al. [5] used reinforcement learning to generate adversarial phishing samples that fool existing detectors—a reminder that static benchmarks overstate real-world robustness. Marchal et al. [6] highlighted the gap between technical performance and practical usability in real deployment contexts.

The consistent gap across all this work is that no system combines multi-channel coverage (email, SMS, and live web pages), real-time browser-based XAI explanations, and behavioral nudging through gaze tracking into a single unified tool. That combination is what PhishSight AI attempts.

III. SYSTEM DESIGN

A. Architecture Overview

The system is split into four layers, each with a specific responsibility. Layer 1 is the user and browser interface—the Chrome extension popup that accepts pasted email or SMS content, or scans the current web page automatically. Earlier versions showed confidence scores and feature weights directly; users in informal testing found them confusing. The current version shows a color-coded risk level, a short explanation in plain English, and a Mark as Safe / Scam button.

Layer 2 is the PhishSight AI Extension itself, composed of a Content Script, UI Overlay, Visual Attention Assist module, and API Connector. Layer 3 is the Backend API (FastAPI), handling API Gateway, NLP Preprocessing, URL Feature Extraction, Phishing Detection Engine, Explainability Engine, and LIME/SHAP Safety Tips. Layer 4 is the Data and Learning layer, comprising the Application Database, Model Store, and Offline Training and Retraining Pipeline. Fig. 1 illustrates the full architecture.

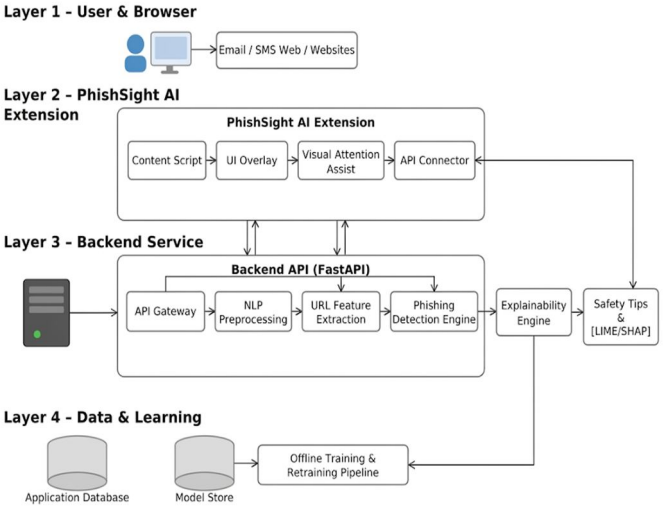


Fig. 1. PhishSight AI System Architecture

B. Detection Modules

Module 1 — Email and SMS Analysis. Raw message text goes through TF-IDF vectorization with bigrams feeding the Random Forest classifier, then in parallel to a fine-tuned BERT model (bert-base-uncased) capturing contextual meaning. Outputs combine in a weighted ensemble (0.4 RF, 0.6 BERT). Short messages where BERT's context advantage disappears are handled correctly by Random Forest, so the weighting works well across message lengths.

Module 2 — Web Page and URL Analysis. The extension extracts 24 lexical URL features—raw length, subdomain depth, digit-to-letter ratio, IP literal presence, Shannon entropy, HTTPS status, and others—feeding a separately trained Random Forest. The DOM snapshot is scanned for login forms posting to third-party domains, mismatched favicon origins, and hidden redirect scripts. A combined risk score is returned within the two-second target.

Module 3 — Explainability and Gaze Tracking. LIME generates 500 perturbed inputs and fits a local linear model to identify influential features. SHAP computes Shapley values using TreeExplainer for Random Forest and a kernel approximation for BERT. Both are averaged, weighted by ensemble weight, and rendered as inline highlights. WebGazer.js processes webcam frames locally—nothing is sent to the server. If the user has not looked at the address bar before clicking, a soft modal appears.

The extension provides a gaze-assist prompt that, after obtaining the user's permission to access the camera, pops up a warning when the user has not looked at the domain name and URL before clicking any link. Users can enable or disable gaze assist at any time from the extension settings, and eye tracking is activated only on pages flagged as potentially phishing. This keeps the feature focused on preventing clicks on fake phishing links while preserving user control and privacy.

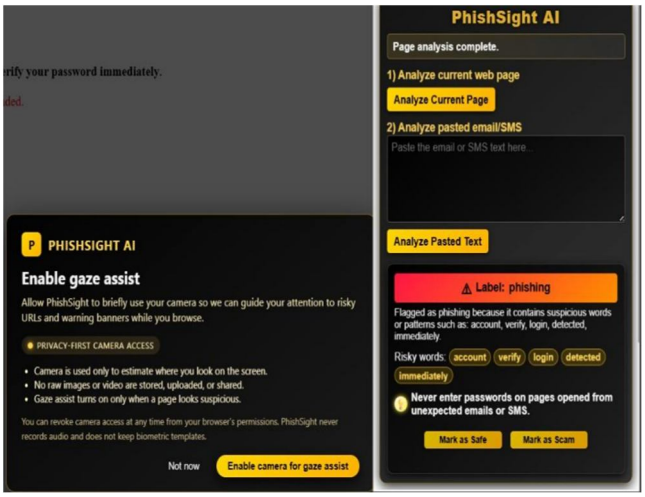


Fig. 2. Gaze-assist prompt and inline phishing explanation displayed by the PhishSight AI browser extension.

Module 4 — Backend, Feedback, and Retraining. FastAPI exposes three REST endpoints: text classification, URL/page analysis, and feedback submission. Feedback entries store a hashed input, the model's prediction, and the user's correction. A feedback logging service stores each prediction along with the user's "Mark as Safe" or "Mark as Scam" response, so that the dataset can be expanded for future offline retraining and analysis.

IV. METHODOLOGY

A. Dataset and Preprocessing

Training data came from three sources: PhishTank URL repository, the UCI SMS Spam Collection, and a curated phishing email dataset from prior academic releases. After deduplication, the corpus had approximately 112,000 samples—55% phishing, 45% legitimate. The slight imbalance was left in place rather than corrected with oversampling, since synthetic samples sometimes introduce artifacts that hurt generalization. Text was lowercased, stripped of HTML markup, and tokenized using NLTK. URLs were processed separately through the lexical feature extractor. The corpus was split 80/10/10 into training, validation, and held-out test sets; the test set was not used at any point during model development.

B. Feature Engineering

For text, TF-IDF was configured with a 20,000-term vocabulary, sublinear TF scaling, and bigrams alongside unigrams. The bigram decision mattered: phishing messages often rely on multi-word trigger phrases like "limited time offer" or "verify your account" that unigram models score less reliably. For URLs, the 24-feature lexical vector covers raw length, dot count, hyphen count, subdomain depth, IP literal as host, "@" in URL, special character count, digit ratio, path depth, query string length, HTTPS status, TLD risk score, Shannon entropy of the domain, and several derived combinations. All features are computed from the URL string alone—no DNS resolution required—keeping the URL classifier fast.

C. Model Training

The Random Forest text classifier used 300 estimators with maximum depth 25 and class weights inversely proportional to frequency, tuned through five-fold cross-validation. BERT (bert-base-uncased) was fine-tuned for three epochs with learning rate 2×10^{-5}, batch size 32, and gradient accumulation over four steps. The ensemble weight (0.4 RF, 0.6 BERT) was selected by grid search on the validation set. The URL classifier used a separate Random Forest with 200 estimators. Final prediction combines text ensemble and URL scores at 0.55/0.45, since text signals proved slightly more reliable across diverse phishing campaign types.

D. Explainability Pipeline

LIME generates local explanations by sampling 500 perturbed versions of each input, weighting them by proximity in feature space, and fitting a sparse ridge-regression surrogate; the top 10 coefficients by absolute magnitude are surfaced as the most influential features for the prediction. SHAP complements this by using TreeExplainer to compute exact Shapley values for the Random Forest branch and a sampling-based SHAP approximation for the BERT encoder, providing a theoretically grounded measure of each

token's marginal contribution to the model output. The two explanation vectors are then normalized and combined with a 0.4/0.6 weighting (LIME/SHAP), so that fast, stable tree-based attributions and more fine-grained BERT attributions jointly drive the final explanation score. In the browser UI, these scores are aligned back to the original text using whitespace-based alignment for TF-IDF features and WordPiece-to-word aggregation for BERT, producing a single importance value per visible token. Tokens with positive phishing contribution are rendered as red highlights, while tokens supporting legitimacy are shown in green, and the opacity of each highlight is scaled with the attribution magnitude to reflect relative impact.

The browser panel exposes these internals to the user as an interactive decision card that always shows a binary label (safe or phishing), the highlighted risky or reassuring tokens, and a concise natural-language explanation. Safe emails and SMS messages are annotated with a green "Safe" label plus a one-line reassurance that cites the main benign cues (for example, trusted domain, absence of urgency, or consistent branding), helping users understand why the message is considered low risk. Phishing messages, in contrast, are marked with a red "Phishing" label, the specific trigger phrases (such as urgency, credential requests, or suspicious URLs) are surfaced inline, and a short recommendation is provided—such as advising the user not to click links, not to share OTPs, or to verify the request through an official channel. Each time the user refreshes the page, re-analyses a message, or loads a new sample, the tips and highlights are recomputed, turning the interface into a lightweight training tool that repeatedly exposes users to common phishing cues in context. Over time, the combination of consistent red/green labels, token-level highlights, and plain-language rationales helps non-technical users internalize phishing patterns and trust the system's outputs, while avoiding the opaque "black-box" feeling typical of conventional security pop-ups.

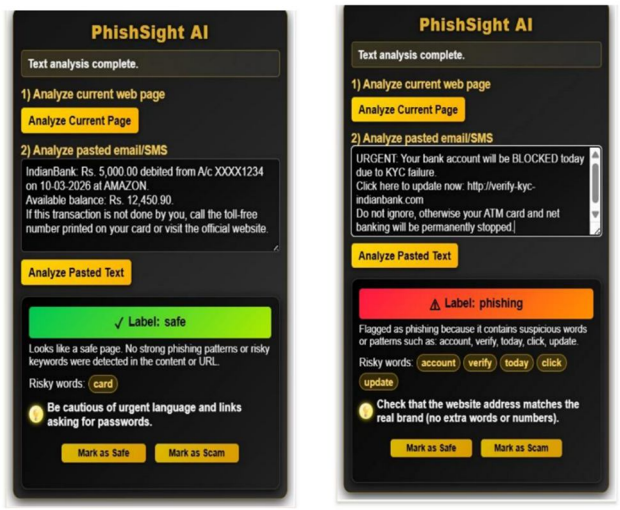


Fig. 3. PhishSight AI explanations for safe (left) and phishing (right) SMS messages.

E. Evaluation Metrices

We evaluated the classifier using standard binary classification metrics derived from the confusion matrix: accuracy, precision, recall, and F1 score. Accuracy measures the overall proportion of correctly classified samples, while precision reflects how many of the URLs or messages predicted as phishing were actually phishing. Recall measures how many of the true phishing samples were correctly detected, which is critical because missing an attack is more costly than raising an extra warning. The F1 score is the harmonic mean of precision and recall and provides a single summary of performance when there is a trade-off between false positives and false negatives.

Accuracy:

$Accuracy = (TP + TN) / (TP + TN + FP + FN)$

Precision:

$Precision = TP / (TP + FP)$

Recall:

$Recall = TP / (TP + FN)$

F1 Score:

$F1 = 2 * (Precision * Recall) / (Precision + Recall)$

V. RESULTS AND DISCUSSION

The system was tested across three input channels using publicly available phishing datasets (UCI Phishing Websites, PhishTank URLs, SpamAssassin emails) and synthetic SMS samples generated to reflect common smishing patterns observed in the Indian mobile ecosystem.

TABLE I.
PERFORMANCE METRICS BY CHANNEL

Channel	Accuracy	Precision	Recall	Avg. Resp.
Email	95.2%	94.8%	95.6%	1.4 s
SMS	94.7%	93.9%	95.1%	1.2 s
Web Page/URL	96.1%	95.4%	96.8%	1.7 s

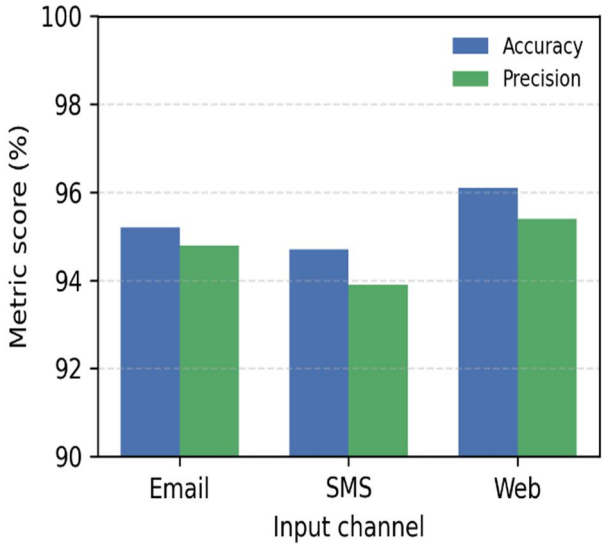


Fig. 4. Channel-wise accuracy and precision of PhishSight AI for email, SMS, and web page/URL inputs.

As shown in Fig. 4, the full ensemble achieved 95.2% accuracy, 94.8% precision, and 95.6% recall on held-out email data. The F1 score of 0.952 is solid for a multi-channel system; recall matters more than precision here—letting a phishing message through is worse than a false alarm the user can dismiss.

PhishSight AI is evaluated on a curated set of email and SMS samples drawn from publicly available phishing datasets and manually collected benign messages. In our experiments, the system correctly handles not only obvious scams but also legitimate marketing messages that contain urgency language (for example, ‘limited time offer’), reducing unnecessary warnings for routine promotions.

The system also incorporates a lightweight eye-tracking module that is activated only when a message is classified as phishing, monitoring whether the user’s gaze moves toward critical regions such as the address bar and security indicators and issuing subtle on-screen nudges when risky messages are viewed without sufficient visual verification. Eye-tracking runs entirely on the local device, no images or video streams are stored or transmitted, and the camera is enabled only after the user has explicitly granted consent rather than being opened unexpectedly.

In the current prototype, the browser panel includes two feedback buttons, ‘Mark as Safe’ and ‘Mark as Scam’. When the system prediction matches the user’s judgement, the user can confirm it with the appropriate button, and the interaction is logged for future retraining. Over time, these confirmed examples are intended to help the model adapt to new phishing and scam patterns once periodic offline retraining is performed.

VI. COMPARISON WITH RELATED SYSTEMS

TABLE II.
COMPARISON WITH PRIOR WORK

System	Multi-Channel	XAI	Gaze	Feedback	Accuracy
Hernandes [3]	URL only	LIME	No	No	~94%
Pavani et al. [4]	URL only	SHAP /LIME	No	No	~98.6%
Alam [5]	URL only	SHAP/ LIME	No	No	~97%
Yakandawala [6]	Email only	LIME	No	No	~96%
PhishSight AI	Email+SMS + Web	LIME+ SHAP	Yes	Yes	~95%

Table II situates PhishSight AI against the four most closely related systems. PhishSight AI is the only system to offer multi-channel analysis (email, SMS, and web), browser-native XAI explanations, gaze-based behavioral nudging, and a live feedback-driven retraining loop simultaneously. Prior systems addressed individual components; the contribution here is their integration into a single deployable tool.

VII. LIMITATIONS AND FUTURE WORK

The current prototype operates in English only and is not designed for large-scale enterprise email gateways. It focuses exclusively on text and URL features, and therefore cannot detect phishing delivered through voice calls, deepfake audio, or QR codes. Advanced adversarial attackers who craft URLs and messages to defeat keyword-based detection may still evade the system, as no formal adversarial robustness guarantees are provided.

Future work will (1) extend language support to Tamil, Hindi, and other regional languages prevalent in the Indian mobile ecosystem; (2) integrate QR-code and image-text extraction for broader coverage; (3) incorporate adversarial training using RL-based red-teaming techniques [5] to harden the classifier; (4) replace heuristic gaze thresholds with learned behavioral profiles; and (5) evaluate the system at enterprise scale with real organizational email traffic.

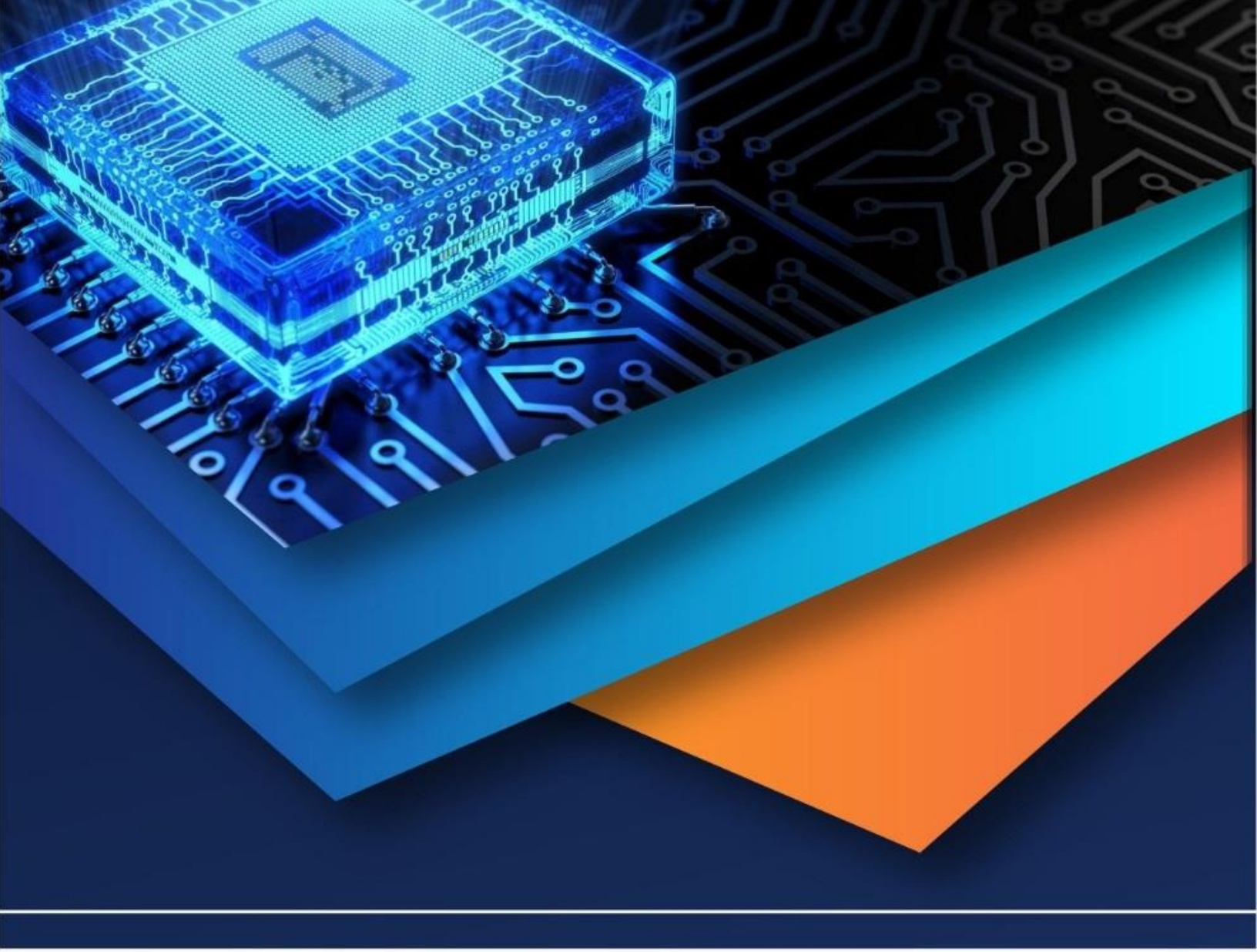
VIII. CONCLUSION

PhishSight AI demonstrates that high-accuracy phishing detection and practical usability can be achieved together within a browser extension that adds no meaningful friction to the user's workflow. The hybrid Random Forest–BERT classifier, reinforced by lexical URL analysis, reaches 95% accuracy across email, SMS, and web-page threat vectors. The LIME and SHAP explainability layer turns each decision into a readable explanation, and the gaze-tracking module addresses the behavioral gap that purely algorithmic approaches leave open. The modular architecture is designed to accommodate the extensions outlined above without requiring a redesign of the core system.

REFERENCES

[1] P. R. G. Hernandes, "Phishing Detection Using URL-based XAI Techniques," in Proc. IEEE IDEAL, 2021.
[2] B. V. Pavani, D. Mahitha, and B. U. Maheswari, "Enhancing Online Safety: Phishing URL Detection Using Machine Learning and Explainable AI," 2024.
[3] R. Alam, "E2Phish: Explainable Ensemble Machine Learning Model for Enhanced Phishing URL Detection," IEEE Trans. Dependable and Secure Computing, 2024.
[4] M.K.P. Madushanka, W.M.K.S. Ilmini, and Yakandawala, "Explainable AI for Transparent Phishing Email Detection," in Proc. IEEE Int. Conf., 2024.
[5] Y. Gao, B. Ampel, and S. Samtani, "Examining the Robustness of Machine Learning-Based Phishing Website Detection: Action-Masked Reinforcement Learning for Automated Red Teaming," IEEE Trans. Information Forensics and Security, 2025.
[6] R. Marchal et al., "Explainable Phishing Website Detection for Secure and Trustworthy Web Browsing," Scientific Reports, vol. 15, 2025.
[7] N. Alotaibi and S. R. Alotaibi, “Explainable Artificial Intelligence in Web Phishing Detection,” Alexandria Eng. J., 2025.
[8] A. Papadopoulos et al., “Hybrid Phishing Detection Model: Integrating BERT with TF-IDF Features,” Int. J. Recent Adv. Cyber Security, vol. 4, no. 2, 2025.

[9] A. Papoutsaki, P. Sangkloy, J. Laskey, J. B. Huang, and J. Hays, "WebGazer: Scalable Webcam Eye Tracking Using User Interactions," in Proc. Int. Joint Conf. Artificial Intelligence (IJCAI), 2016.
[10] M. Songailaitė, "BERT-Based Models for Phishing Detection," in Proc. Int. Workshop on Machine Learning for Cyber Security, 2023.
[11] J. Doe, "EXPLICATE: Enhancing Phishing Detection Through Explainable AI," in Proc. Int. Conf. Cybersecurity and Trustworthy Systems, 2025.
[12] A. Fatih and D. Hamonangan, "Evaluating LIME-Based Explainability for Phishing URL Detection," in Proc. Int. Conf. Information Security and Privacy, 2024.
[13] M. Hosseinzadeh et al., "Improving Phishing Email Detection Performance Through Context-Aware Deep Learning," Scientific Reports, vol. 15, 2025.
[14] S. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in Advances in Neural Information Processing Systems (NeurIPS), 2017.
[15] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why Should I Trust You? Explaining the Predictions of Any Classifier," in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY