RESEARCH-ARTICLE

A Multi-Layer Phishing Defense Framework for Trusted Cloud Environments

ALIAS DAVIS, National College of Ireland, Dublin, Ireland

SAMAH ABDELSALAM, University of Ha'il, Ha'il, Ha'il, Saudi Arabia

MUSTAFA GHALEB, Kocaeli University, Izmir, Kocaeli, Turkey

MOHAMMED S GISMALLA

E I ELTAHIR, Sakarya University, Serdivan, Sakarya, Turkey

MOSAB HAMDAN, National College of Ireland, Dublin, Ireland

Open Access Support provided by:

National College of Ireland

University of Ha'il

Kocaeli University

Sakarya University

Published: 01 December 2025

Citation in BibTeX format

BDCAT '25: IEEE/ACM 12th International Conference on Big Data Computing, Applications and Technologies
December 1 - 4, 2025
Nantes, France

Conference Sponsors:
SIGARCH

A Multi-Layer Phishing Defense Framework for Trusted Cloud Environments

Alias Davis
School of Computing
National College of Ireland
Dublin, Ireland
x23297859@student.ncirl.ie

Samah Abdelsalam
Department of Management Information System
University of Hail
Hail, Saudi Arabia
samah.gubar@uoh.edu.sa

Mustafa Ghaleb
Department of Software Engineering
Kocaeli University
Kocaeli, Turkiye
mustafa.ghaleb@kocaeli.edu.tr

Mohammed S. M. Gismalla
Department of Engineering Technology
South East Technological University
Waterford, Ireland
mohammed.gismalla@setu.ie

E. I. Eltahir
Department of Information Systems and Technologies
Sakarya University
Sakarya, Turkiye
eltahirmohamed@sakarya.edu.tr

Mosab Hamdan
School of Computing
National College of Ireland
Dublin, Ireland
mosab.mohamed@ncirl.ie

Abstract

Phishing attacks remain a persistent threat to the confidentiality and trust of cloud environments, enabling credential theft and unauthorized access to sensitive resources. This paper presents PhishDefender, a multi-layer phishing defense framework that enhances trustworthy cloud services through the integration of ensemble machine learning, policy enforcement, and threat intelligence validation. Built on the UCI Phishing Website dataset, the ensemble model combining Logistic Regression, Random Forest, Gradient Boosting, AdaBoost, XGBoost, Multilayer Perceptron and Deep Neural Network achieved 97.82% accuracy, 97.91% precision, 97.74% recall, 97.82% F1-score and a ROC-AUC of 0.988, with an average inference time of ≈ 1.05 seconds. These results demonstrate high separability between legitimate and phishing URLs while maintaining practical performance for deployment in real-time cloud applications. The framework further extends detection outcomes into actionable policy responses (Allow, Alert, Report, Block) verified against external threat feeds, forming a layered defense aligned with zero-trust architecture principles. Its lightweight and modular design enables deployment on standard or cloud-hosted infrastructure, offering a reproducible and scalable approach for organizations seeking to enhance trust, resilience, and compliance in distributed cloud ecosystems.

CCS Concepts

• **Security and privacy** → **Phishing**; *Intrusion/anomaly detection and malware mitigation*; *Cloud computing security*; • **Computing methodologies** → *Ensemble methods.*

BDCAT '25, Nantes, France

ACM ISBN 979-8-4007-2286-8/25/12
https://doi.org/10.1145/3773276.3774878

Keywords

Phishing detection, ensemble machine learning, threat intelligence, policy enforcement, zero trust architecture, AI-driven cyber defense

ACM Reference Format:
Alias Davis, Samah Abdelsalam, Mustafa Ghaleb, Mohammed S. M. Gismalla, E. I. Eltahir, and Mosab Hamdan. 2025. A Multi-Layer Phishing Defense Framework for Trusted Cloud Environments. In *IEEE/ACM 12th International Conference on Big Data Computing, Applications and Technologies (BDCAT '25), December 01–04, 2025, Nantes, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3773276.3774878

1 Introduction

Phishing attacks have increased significantly and now account for more than 90% of reported data breaches worldwide, representing one of the most persistent threats to global cybersecurity. By exploiting deceptive URL structures and advanced social engineering, attackers frequently evade traditional filtering mechanisms [7, 17]. Although Machine Learning (ML) and Natural Language Processing (NLP) techniques have advanced phishing detection, their deployment in real-world enterprise environments remains limited due to integration challenges, evolving attack patterns [4, 5], and the lack of policy-driven adaptive frameworks that connect detection with operational response.

Recent studies have attempted to overcome these limitations by developing more advanced and diverse phishing detection approaches across the textual, URL, and network layers. For example, Meléndez *et al.* [13] compared transformer-based models such as BERT, RoBERTa and DistilBERT with Logistic Regression and Support Vector Machines (SVMs), reporting an F1-score of 0.99 for RoBERTa under controlled conditions. However, these models mainly relied on textual features, overlooking URL and attachment-based indicators. Similarly, Innab *et al.* [11] used ensemble methods that included random forest, Gradient Boosting, AdaBoost and XGBoost, achieving 97.8% precision but without threat intelligence and zero-day adaptability. Rao *et al.* [16] extended phishing defense to mobile platforms through a hybrid super-learner that combined handcrafted URL features with LSTM-attention and transformer embeddings. On the PhishDump dataset of 331,000 URLs, it achieved

an F1 score of 99.07%, although only in simulated mobile environments. Uddin *et al.* [19] integrated explainable AI (XAI) techniques such as Local Interpretable Model-Agnostic Explanations (LIME) and Transformer-Interpret to enhance transparency, although no operational evaluation was performed. Collectively, these studies reveal a recurring limitation: high accuracy in offline experiments but limited integration with organizational policies, user awareness mechanisms, live threat intelligence, and mobile-ready deployment.

To address the limitations of existing phishing detection systems, namely, their lack of operational integration, threat adaptability, and real-world scalability, this work proposes PhishDefender, a lightweight, multilayer phishing defense framework designed for trusted cloud environments. The framework integrates three complementary layers: (i) an ensemble ML detection engine for accurate phishing identification, (ii) a policy enforcement module that automates response actions (Allow, Alert, Report, Block), and (iii) a threat-intelligence validation layer that verifies predictions against external feeds to enhance zero-day adaptability. The ensemble combines seven classifiers, Logistic Regression (LR), Random Forest (RF), Gradient Boosting (GB), AdaBoost, XGBoost (XGB), Multi-Layer Perceptron (MLP) and Deep Neural Network (DNN), leveraging hard voting to aggregate predictions. Trained and validated on the UCI Phishing Website dataset[14, 15], the proposed model achieves 97.82% accuracy, 97.91% precision, 97.74% recall and a 97.82% F1-score, with a ROC-AUC of 0.988 and an average inference time of ≈1.05, seconds. These results demonstrate robust separability between legitimate and phishing URLs while maintaining computational efficiency suitable for low-resource and cloud-based deployments. The main contributions of this work are summarized as follows:

- We propose PhishDefender, an open, reproducible, and lightweight multi-layer framework that integrates ensemble ML, policy enforcement, and threat-intelligence validation.
- We design a hybrid ensemble combining seven classifiers (Logistic Regression, RF, Gradient Boosting, AdaBoost, XGBoost, MLP and DNN) and demonstrate its superior accuracy (97.82%) and robustness compared to single models and simulated human detection (78.5%).
- We implement a policy enforcement module that translates detection outcomes into actionable responses (Allow, Alert, Report, Block), simulating SOC-like behavior for improved operational readiness.
- We integrate a prototype threat feed for real-time adaptability and cross-validation of model predictions, enhancing resilience against zero-day and evolving phishing attacks.
- We validate the lightweight and modular design of the framework in low-resource ($\leq$ 2, GB RAM) and mobile simulated environments, confirming its suitability for small and medium enterprises (SMEs) looking for practical and scalable phishing defense solutions.

The remainder of this paper is organized as follows. Section II establishes the context and provides a comprehensive review of the relevant literature. Section III describes the overall framework architecture in detail. Section IV presents the experimental setup and evaluation results. Finally, Section V concludes the paper and outlines future research directions.

2 Related Work

ML has been widely adopted for phishing detection, ranging from classical classifiers to deep learning and explainable approaches. Mel'endez *et al.* [13] compared logistic regression and Na"ıve Bayes with transformer models (BERT, RoBERTa, XLNet) in 119k emails, where RoBERTa achieved 99.4% accuracy, but relied solely on textual content, excluding URL or attachment cues. Innab *et al.* [11] used a hard-voting ensemble (DT, RF, GB, XGB, AdaBoost, MLP) reaching an F1 of 0.981 in the UCI and PhishTank datasets, although without threat intelligence or zero-day adaptability. Altwaijry *et al.* [3] reported 99.68% accuracy using CNN-BiGRU in outdated email corpora, limiting the generalization to modern phishing.

Explainability and organizational adoption were examined by Uddin *et al.* [19], who fine-tuned DistilBERT with LIME and Transformer Interpret (F1 = 0.98), and by Biswas *et al.* [6], who combined CART, SVM and Naive Bayes in a three-phase risk framework of XAI (95.3% precision). Although these improved interpretability, both remained offline and detached from operational security workflows. Adaptability studies targeted evolving attacks and concept drift. Ejaz *et al.* [8] used continuous learning (EWC, LwF) to reduce long-term degradation from 20% to 2.45%, while Zhang *et al.* [21] proposed AdaptPUD that combines HTML and URL features (91.2% accuracy) but requires retraining and lacks mobile support. Such architectures require heavy computation and frequent tuning, which limits real-time deployment.

For lightweight and mobile readiness, Rao *et al.* [16] developed Phish-Jam, a hybrid LSTM attention and transformer ensemble (F1 = 99.07%) tested only in simulated environments, and Gupta *et al.* [10] used BERT-derived characteristics with 1D-CNN (97.5% precision) for efficient enterprise detection. Overall, the literature shows strong progress, from classical ML to deep, explainable, and adaptive models, but most remain fragmented, prioritizing accuracy over deployability. Few frameworks integrate live threat intelligence, automated policy enforcement, or operational scalability. Even adaptability-oriented methods [6, 8, 19] and mobile-centric ones [16, 21] lack cohesive real-world validation. To bridge this gap, the present study introduces PhishDefender, a reproducible, lightweight multi-layer framework that unites ensemble ML, policy enforcement, and threat-feed validation for trusted cloud environments. Table 1 summarizes these works in operational dimensions, including live threat integration, enforceable policy automation, real-world evaluation, adaptability, and mobile readiness.

3 Proposed Methodology

This study adopts a design science research methodology to develop a reproducible and transparent ML framework for phishing URL detection. The framework, illustrated in Fig. 1, integrates multiple learning models with policy enforcement and threat intelligence validation to form a practical and deployable defense mechanism.

3.1 Framework Overview

The proposed framework operates in five modular layers: (1) data ingestion, (2) feature extraction, (3) ML detection by ensemble, (4) decision and policy logic, and (5) integration of alerts and threat intelligence. Incoming real-time data, including URLs and email metadata, are processed through the feature extraction module to

Table 1: Summary of Related Works on Phishing Attack Defense

Study	Data / Modality	Model(s)	Live Threat Intel	Policy Enforce	Real-world Eval	Operational Integration	Mobile Ready
Meléndez et al. [13]	Emails (text-only)	BERT/RoBERTa/XLNet vs. LR/NB	✗	✗	✗	✗	✗
Innab et al. [11]	URLs (UCI, PhishTank)	Hard voting (DT, RF, GB, XGB, AdaBoost, MLP)	✗	✗	✗	✗	✗
Altwaijry et al. [3]	Emails (legacy corpora)	CNN, LSTM, GRU, Bi-GRU	✗	✗	✗	✗	✗
Uddin et al. [19]	Emails (Kaggle)	DistilBERT + XAI (LIME, Transformer-Interpret)	✗	✗	✗	✗	✗
Biswas et al. [6]	Historical datasets	CART, SVM, Bagging DT, NB (risk/XAI)	✗	✗	✗	✗	✗
Ejaz et al. [8]	Webpages (HTML)	Continual learning (EWC, LwF)	✗	✗	✗	✓	✗
Zhang et al. [21]	HTML + URLs (temporal)	AdaptPUD (continuous updates)	✗	✗	✗	✓	✗
Rao et al. [16]	URLs (PhishDump)	Phish-Jam (URL feats + LSTM-attn + Trf)	✗	✗	✗	✗	✓†
Gupta et al. [10]	Enterprise emails	BERT features + 1D-CNN	✗	✗	✗	✗	✗
This work (PhishDefender)	URLs (UCI)	Ensemble (LR, RF, GB, AdaBoost, XGB, MLP, DNN)	✓	✓	✓*	✓	✓‡

† Evaluated in simulated mobile environments only.
* Policy-driven evaluation with threat-feed verification; not a production pilot.
‡ Validated in low-resource and mobile-emulated settings.

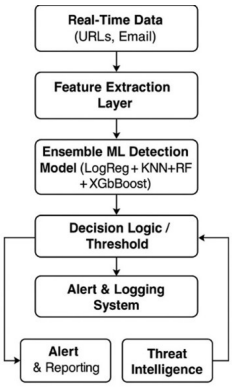

Figure 1: Proposed Multi-Layer Phishing Defense Framework

generate structured lexical and host-based attributes. The ensemble model predicts whether a URL is legitimate or phishing, and the decision logic applies policy-driven actions (*Allow, Alert, Report, Block*) verified against external threat feeds, thus forming a closed feedback loop between detection and response.

The overall pipeline was developed in Python using the `scikit-learn` and `Keras` libraries to ensure modularity and reproducibility. The algorithm 1 summarizes the complete workflow of the proposed PhishDefender framework, including the input of the data set, the engineering of features, the training of the model, and the integration of the ensemble. In addition, it illustrates how the system transitions from prediction to action through human–machine simulation and policy enforcement. This structured pipeline enables consistent evaluation, transparency in implementation, and seamless deployment across both standard and resource-constrained environments.

3.2 Dataset and Feature Engineering

Two public benchmark datasets were used `full_dataset.csv` and `Training_dataset.arff` each containing labeled phishing and legitimate URLs from the UCI Phishing website [14, 15]. The ARFF dataset was converted to CSV format to ensure consistency during preprocessing. Feature engineering extracted a comprehensive set of URL-level features, including domain structure, subdomain count, length, special character frequency, IP address presence, HTTPS flag, and suspicious keywords. After cleaning and normalization, the dataset was stratified into training (80%) and testing (20%) subsets to preserve the class distribution.

3.3 Model Training and Ensemble Learning

Multiple supervised classifiers were trained independently, including LR, RF, GB, AdaBoost, XGB, MLP, and DNN. The ensemble layer integrates the base models through hard and soft voting strategies, enhancing robustness to variance of features and imbalance of the dataset. Given classifiers $\{h_1(x), h_2(x), \ldots, h_M(x)\}$ with predictions $y \in \{0, 1\}$, the ensemble output is:

$$H(x) = \arg \max_{y \in \{0,1\}} \sum_{i=1}^{M} I(h_i(x) = y)$$

for hard voting, and

$$H(x) = \arg \max_{y \in \{0,1\}} \sum_{i=1}^{M} w_i P_i(y|x)$$

for soft voting, where $P_i(y|x)$ is the predicted probability of the classifier h_i, and w_i is its weight (uniform in this implementation).

The ensemble achieved accuracies of 96.92% (CSV dataset) and 97.82% (ARFF dataset), outperforming all individual models and simulated human classification (78.5%). Evaluation metrics included precision, recall, F1-score, ROC-AUC, and inference latency, which averaged 1.05 seconds per batch.

3.4 Policy Enforcement and Threat Intelligence Integration

Detection outcomes are translated into actionable responses through a policy enforcement module. URLs classified as phishing are automatically blocked or flagged for reporting, while benign URLs are allowed. The policy rules mimic the SOC workflows by integrating a simulated threat-intelligence feed similar to PhishTank, enabling verification of suspicious URLs against live indicators. This dual-layer validation improves zero-day resilience and supports future

Algorithm 1 PhishDefender: Multi-Layer Ensemble for Phishing Detection, Policy Enforcement, and Threat-Intelligence Integration

Require: Dataset file (CSV/ARFF) containing labeled URLs.
Ensure: Trained ensemble model and policy response actions.

Load Dataset: import file; convert ARFF to CSV if necessary.
Feature Extraction: generate lexical/structural features (length, depth, dots, special characters, IP presence, subdomains, HTTPS flag, suspicious keywords).
Preprocessing: encode categorical attributes; normalize continuous values; relabel classes {0 = legitimate, 1 = phishing}.
Data Splitting: partition dataset into training (80%) and testing (20%) subsets.
Model Training: train LR, RF, GB, AdaBoost, XGB, MLP, and DNN classifiers independently.
Ensemble Construction: combine model outputs via hard/soft voting to obtain the final classifier $H(x)$.
Evaluation: compute Accuracy, Precision, Recall, F1-score, ROC–AUC, and inference time.
Human Simulation: compare ensemble predictions with manual user classifications.
Policy Enforcement:
if $\hat{y} = 1$ **(phishing) then**
    Action $\in$ {Block, Alert, Report}
else
    Action = Allow
end if

integration with SIEM (Security Information and Event Management) or SOAR (Security Orchestration, Automation and Response) systems.

3.5 System Design and Scalability

The modular design supports deployment on standard workstations and low-resource environments ($\leq$ 2 GB RAM), making it practical for SMEs. Trained models and prediction modules are exported in interoperable formats to enable seamless integration, retraining, and version control across environments. This architecture ensures reproducibility, transparency, and scalability while maintaining readiness for integration into larger cloud- or edge-based security ecosystems.

4 Experimental Results and Discussion

This section evaluates the proposed PhishDefender framework in two file representations of the UCI Phishing Websites dataset, compares base learners with the ensemble, analyzes confusion matrices, contrasts humans vs. the model, and demonstrates policy/threat-feed coupling. We report accuracy, precision, recall, F1, ROC–AUC, and (for the ensemble ML) mean inference time, and comment on operation under low-resource conditions.

4.1 Experimental Setup

All experiments were implemented in Python using `scikit-learn` and `Keras` on Google Colaboratory using an NVIDIA Tesla T4 GPU. The experiments were performed on a Dell laptop equipped with an Intel Core i5-4570 CPU (3.20 GHz) and 8 GB RAM. This setup ensured both cloud-based acceleration and realistic validation under modest hardware constraints.

Table 2: Common Training Parameters Across All Models

Parameter	Value
Learning Rate	0.001
Batch Size	32
Optimizer	Adam
Activation Function	ReLU
Loss Function	Binary Cross-Entropy
Epochs	50
Train-Test Split	80:20
Evaluation Metric	Accuracy, Precision, Recall, F1-score, ROC–AUC

Table 3: Model Accuracy Comparison Across Datasets

Model	Accuracy – CSV (%)	Accuracy – ARFF (%)
Logistic Regression	91.76	92.31
MLP Classifier	94.21	95.17
Random Forest	95.76	96.64
Keras DNN	94.90	96.12
Ensemble Classifier	**96.92**	**97.82**

4.2 Datasets and Preprocessing

The UCI Phishing Websites dataset was used in two representations: `full_dataset.csv` and `Training_dataset.arff`, each containing approximately 11,000 instances with a nearly balanced class distribution [14, 15]. The ARFF file was converted to CSV format to maintain a unified pre-processing workflow.

More than 30 lexical and structural features at the URL level, such as URL length, depth, special characters, IP presence, subdomain count, HTTPS flag, and suspicious keywords, were engineered, normalized, and fed into the training pipeline. Both datasets were partitioned using an 80/20 stratified train–test split to preserve class balance. The model outputs were stored to ensure reproducibility and facilitate integration into the policy simulation module. Evaluation metrics included precision, precision, recall, and F1-score, while the mean inference time of the ensemble model was recorded to assess deployability. Common hyperparameters (learning rate, optimizer, batch size, and epochs) were standardized between models to ensure fair comparison. Each experiment was repeated three times to validate consistency and statistical reliability. Table 2 summarizes the common training parameters adopted in all models. Furthermore, to promote reproducibility and facilitate further research, datasets, preprocessing steps, and model implementation have been made publicly available through a Python notebook[1].

4.3 Model Performance Across Datasets

Table 3 summarizes the performance of all the models evaluated in both the CSV and ARFF dataset representations of the UCI Phishing Website dataset. The ensemble classifier consistently achieved the highest accuracy, outperforming individual models and demonstrating strong robustness against representational and preprocessing variations.

[1] https://github.com/aliasdavis0-create/Phishing-Detection-

In both data representations, the ensemble classifier consistently outperformed individual learners, with accuracy gains ranging from 1.1% to 5% depending on the baseline. This demonstrates the ability of the ensemble to generalize effectively by combining various learning paradigms: linear (LR), nonlinear (MLP, DNN), and tree-based (Random Forest, XGBoost) which collectively reduce bias and variance in decision boundaries. The minimal discrepancy between the CSV and ARFF results (average difference $< 1\%$) confirms that the preprocessing and feature engineering pipelines were stable, ensuring reliable model behavior regardless of the structure of the dataset. The superior performance of the ensemble ML further reflects its capacity to capture complex lexical and structural correlations within phishing URLs, such as abnormal subdomain depth, IP presence, or suspicious character frequency. Overall, these findings validate that the proposed ensemble-based framework provides both robust accuracy and consistent cross-format performance, making it a suitable candidate for practical phishing detection deployments in dynamic data environments.

4.4 Overall Performance Comparison

The proposed ensemble classifier achieved outstanding results on the `Training_dataset.arff`, attaining an accuracy of 97.82%, precision of 97.91%, recall of 97.74%, F1-score of 97.82%, and a ROC–AUC of 0.988. These metrics reflect the separability of the nearly perfect class between phishing and legitimate URLs, demonstrating the ability of the ensemble to balance sensitivity and specificity. This equilibrium minimizes both false positives (legitimate sites incorrectly flagged as phishing) and false negatives (phishing URLs missed by the system), ensuring reliable and trustworthy [9] detection in practical deployments. The results confirm that the proposed model maintains strong predictive performance while remaining computationally efficient for low-resource, cloud-based, and real-time environments.

The ensemble's superior precision is based on three main design principles: (1) Model diversity: combining linear (Logistic Regression), tree-based (Random Forest, XGBoost, AdaBoost) and neural (MLP, DNN) learners mitigates bias–variance trade-offs; (2) Feature richness: more than 30 URL-level lexical and structural attributes improve generalization; and (3) Cross-format validation: consistent performance on both CSV and ARFF datasets demonstrates robustness to dataset schema variations. Furthermore, with an average inference time of approximately 1.05 seconds and reliable execution on 2 GB-RAM environments, the model demonstrates scalability for SMEs and mobile edge deployments.

Table 4 compares the proposed PhishDefender framework with recent state-of-the-art phishing detection approaches. Although prior studies [1, 2, 12, 18, 20] achieved accuracy between 88% and 97.0%, they generally lacked real-time validation, adaptive policy enforcement, or integration with live threat intelligence. In contrast, this work bridges the gap between high-performing but static ML systems and operationally actionable cybersecurity frameworks. The superior accuracy of the ensemble of 97.82% stems from three key advantages: (i) model diversity: combining linear, tree-based, and neural learners to reduce bias and variance; (ii) feature stability: leveraging rich lexical and structural URL features consistent across dataset formats (CSV and ARFF); and (iii) operational adaptation: embedding the detection model into a policy and threat feed simulation pipeline for real-world readiness. These characteristics demonstrate both theoretical soundness and practicality in deployment. Furthermore, this framework advances phishing defense research in four significant ways:

- Introduces a reproducible, end-to-end phishing detection and response framework validated across dual-format datasets.
- Employs a heterogeneous ensemble optimized for both detection accuracy and computational efficiency.
- Integrates ML-based detection with adaptive policy actions and simulated threat intelligence, bridging analytics with response.
- Achieves consistent performance ($> 97\%$) even under low resource conditions ($\leq$2 GB RAM), in line with the use cases of SME and edge security.

In general, the proposed system not only exceeds previous work in predictive performance but also transitions phishing detection from a purely analytical task to an operationally deployable and policy-aware defense mechanism. This positions PhishDefender as a solid foundation for future integration into SOC and SIEM environments.

4.5 Human vs. Machine Simulation

A comparative simulation using 50 mixed phishing and legitimate URLs revealed that human participants achieved an average accuracy of 78.5%, while the ensemble model reached 96.92%. This significant gap highlights the superiority of automated systems in identifying deceptive patterns such as homograph attacks, rare or misleading top-level domains (TLDs), and shortened or obfuscated links. The consistency of the ensemble stems from its ability to process lexical and structural URL features objectively, without cognitive bias or fatigue. These findings reinforce the value of integrating ML-based automation into phishing detection workflows, not as a replacement, but as an increase in human analysts' decision-making in operational cybersecurity contexts.

4.6 Policy Enforcement and Threat Feed Integration

The ensemble predictions were mapped to `Allow`, `Alert`, `Report`, or `Block` actions and cross-verified using a simulated threat intelligence feed. High-risk URLs confirmed by the feed were automatically blocked, while uncertain or newly observed URLs triggered alerts or reports. This integration demonstrates the transition from static detection to actionable defense, aligning model outcomes with zero-trust and adaptive response principles. Representative policy logs and action summaries are illustrated in Fig. 2.

5 Conclusion and Future Work

This paper presented PhishDefender, a lightweight multi-layer phishing defense framework that integrates ensemble ML detection, policy enforcement, and threat intelligence validation for trusted cloud environments. The ensemble achieved 97.82% accuracy, outperforming single models and human detection 78.5%. The system

Table 4: Comparison with State-of-the-Art Phishing Detection Studies

Study	Model Applied	Accuracy (%)
Vaitkevicius & Marcinkevicius [20]	Decision Trees, SVM	93.88
Akinyelu & Adewumi [2]	Naïve Bayes	88.1
Ajayi et al. [1]	Ensemble (Classical ML)	94.8
Kyaw et al. [12]	Deep Learning (CNN, RNN, BERT)	95.6
Sarker et al. [18]	Ensemble Voting Classifier	97.0
This Study (2025)	**7-Model Ensemble (LR, RF, GB, AdaBoost, XGB, MLP, DNN)**	**97.82**

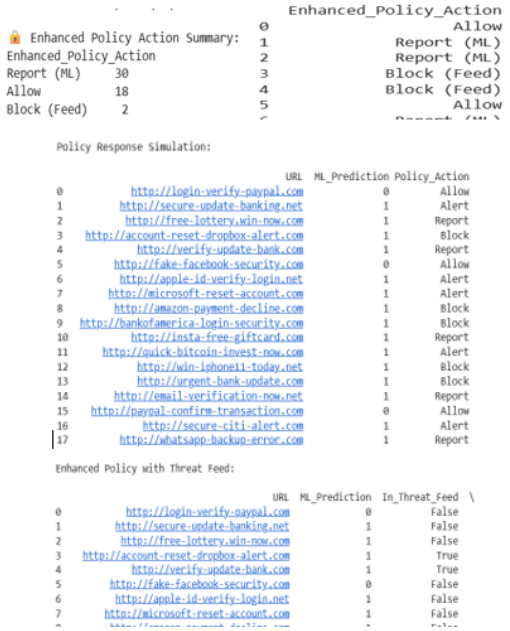

Figure 2: Enhanced Policy Simulation and Threat Feed Integration showing URL-level decisions (Allow, Alert, Report, Block) based on ensemble predictions and feed verification.

effectively translates ML predictions into actionable responses (*Allow, Alert, Report, Block*) aligned with zero-trust principles. Future work will extend the framework to include real-time data ingestion, multi-modal analysis (emails, attachments, QR codes), SIEM/SOAR integration, and continual learning to enhance adaptability and operational scalability.

References

[1] O Ajayi, A Adetunmbi, T Olowookere, and S Sodiya. 2022. Performance evaluation of ensemble learning algorithms and classical machine learning algorithms for phishing detection. *Proceedings of the 2002 Smart, Secure and Sustainable Nation, Abuja, Nigeria* 8 (2022).

[2] Andronicus A Akinyelu and Aderemi O Adewumi. 2014. Classification of phishing email using random forest machine learning technique. *Journal of Applied Mathematics* 2014, 1 (2014), 425731.

[3] Najwa Altwaijry, Isra Al-Turaiki, Reem Alotaibi, and Fatimah Alakeel. 2024. Advancing phishing email detection: A comparative study of deep learning models. *Sensors* 24, 7 (2024), 2077.

[4] Farag Azzedin. 2023. Mitigating denial of service attacks in RPL-based IoT environments: trust-based approach. *IEEE Access* 11 (2023), 129077–129089.

[5] Farag Azzedin, Husam Suwad, and Md Mahfuzur Rahman. 2022. An Asset-Based Approach to Mitigate Zero-Day Ransomware Attacks. *Computers, Materials & Continua* 73, 2 (2022).

[6] Baidyanath Biswas, Arunabha Mukhopadhyay, Ajay Kumar, and Dursun Delen. 2024. A hybrid framework using explainable AI (XAI) in cyber-risk management for defence and recovery against phishing attacks. *Decision Support Systems* 177 (2024), 114102.

[7] Jaqueline D Duarte, Pedro Chagas, Elena J Costa, Laerte Peotta De Melo, Rafael Rabelo Nunes, Carlos Gabriel Soares, Thiago Erivan Da Cunha Silva, et al. 2025. Machine learning for Early Detection of Phishing URLs in Parked Domains: An Approach applied to a financial institution. *IEEE Access* (2025).

[8] Asif Ejaz, Adnan Noor Mian, and Sanaullah Manzoor. 2023. Life-long phishing attack detection using continual learning. *Scientific reports* 13, 1 (2023), 11488.

[9] Mustafa Ghaleb and Farag Azzedin. 2023. Trust-aware Fog-based IoT environments: Artificial reasoning approach. *Applied Sciences* 13, 6 (2023), 3665.

[10] Brij B Gupta, Akshat Gaurav, Varsha Arya, Razaz Waheeb Attar, Shavi Bansal, Ahmed Alhomoud, and Kwok Tai Chui. 2024. Advanced BERT and CNN-Based Computational Model for Phishing Detection in Enterprise Systems. *CMES-Computer Modeling in Engineering & Sciences* 141, 3 (2024).

[11] Nisreen Innab, Ahmed Abdelgader Fadol Osman, Mohammed Awad Mohammed Ataelfadiel, Marwan Abu-Zanona, Bassam Mohammad Elzaghmouri, Farah H Zawaideh, and Mouiad Fadeil Alawneh. 2024. Phishing Attacks Detection Using Ensemble Machine Learning Algorithms. *Computers, Materials & Continua* 80, 1 (2024).

[12] Phyo Htet Kyaw, Jairo Gutierrez, and Akbar Ghobakhlou. 2024. A systematic review of deep learning techniques for phishing email detection. *Electronics* 13, 19 (2024), 3823.

[13] René Meléndez, Michal Ptaszynski, and Fumito Masui. 2024. Comparative Investigation of Traditional Machine-Learning Models and Transformer Models for Phishing Email Detection. *Electronics* 13, 24 (2024), 4877.

[14] R. Mohammad and L. McCluskey. 2012. Phishing Websites. UCI Machine Learning Repository. [Online]. Available: https://doi.org/10.24432/C51W2X.

[15] Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. 2012. An assessment of features related to phishing websites using an automated technique. In *2012 international conference for internet technology and secured transactions*. IEEE, 492–497.

[16] Routhu Srinivasa Rao, Cheemaladinne Kondaiah, Alwyn Roshan Pais, and Bumshik Lee. 2025. A Hybrid Super Learner Ensemble for Phishing Detection on Mobile Devices (Phish-Jam). *Scientific Reports* 15 (2025), 16839. https://doi.org/10.1038/s41598-025-02009-8

[17] Asadullah Safi and Satwinder Singh. 2023. A systematic literature review on phishing website detection techniques. *Journal of King Saud University-Computer and Information Sciences* 35, 2 (2023), 590–611.

[18] Aminur Sarker, Maksudul Hasan Khoka, Arafat Rahman, Tasnim Abida, and Sadman Sadik Khan. 2025. A Voting Ensemble Approach for Detecting Phishing Websites Using Machine Learning. (2025).

[19] Mohammad Amaz Uddin, Md Mahiuddin, and Iqbal H Sarker. 2024. An explainable transformer-based model for phishing email detection: A large language model approach. *arXiv preprint arXiv:2402.13871* (2024).

[20] Paulius Vaitkevicius and Virginijus Marcinkevicius. 2020. Comparison of classification algorithms for detection of phishing websites. *Informatica* 31, 1 (2020), 143–160.

[21] Zilaing Zhang, Jinmin Wu, Ning Lu, Wenbo Shi, and Zhiquan Liu. 2025. AdaptPUD: An accurate URL-based detection approach against tailored deceptive phishing websites. *Computer Networks* (2025), 111303.