Journal of Cybersecurity and Information Management

♦ ISSN: 2690-6775 Vol: 18, No. 01, 2026 ♦ PP. 27–33

Article DOI: https://doi.org/10.54216/JCIM.180104

FULL LENGTH ARTICLE

OPH-Guard: An Operationally Interpretable Tree-Ensemble Framework for Phishing URL Screening in Secure Web Access Management

Reem Atassi[1,*]

[1]Higher Colleges of Technology, UAE

Emails: ratassi@hct.ac.ae

Received: January 10, 2026 Revised: February 11, 2026 Accepted: March 25, 2026 ⋆ Corresponding author

ABSTRACT

Phishing URLs still present a security threat to organizations because they enable credential theft, account takeover, payment fraud, and unauthorized digital service access. Although phishing detection has been studied extensively, many published papers still emphasize predictive performance more than operational system capability, testing, governance, and implementation. This paper develops OPH-Guard as an operational security framework that uses compact tree ensembles to identify phishing URLs for secure web access management. The integrated workflow enables institutional and small-enterprise settings to implement public data ingestion, feature validation, tabular model learning, post-hoc explanation, and security-action mapping. The empirical evaluation used a public GitHub-hosted phishing URL dataset containing 11,481 labeled records and 87 predictive features. A comparison was conducted between Decision Tree, Random Forest, and Extra Trees classifiers under a stratified 80/20 hold-out protocol. Extra Trees produced the best held-out results, with accuracy of 0.9856, precision of 0.9921, recall of 0.9791, F1-score of 0.9855, and ROC-AUC of 0.9984. The study investigates the security relevance of top predictors, including google index, page rank, domain age, and phish hints, showing that the model can support browsing-risk management through URL triage and secure information-management controls. The paper presents a reproducible framework, a complete screening algorithm, a synthesis of ten related studies, and a system that connects model results to security operations.

Keywords: Phishing URL detection ▪ Tree ensembles ▪ Extra Trees ▪ Secure web access management ▪ Operational interpretability ▪ Cybersecurity analytics

1. INTRODUCTION

Cybercriminals use phishing attacks because this attack method requires minimal expense while giving attackers opportunities to steal credentials, compromise accounts, conduct payment fraud, and gain unauthorized access to online platforms. Phishing attacks have developed in two directions: criminals now use more sophisticated technical methods, and their delivery channels continue to shift toward new techniques. For this reason, every organization needs URL screening because it protects users against phishing and other internet threats encountered while accessing online platforms.

Deceptive URLs act as technical elements that can lead to multiple security breaches involving identity-verification systems, customer-access points, financial networks, and cloud-based services. In practical settings, deceptive URLs can disrupt access to web browsers, remote-work systems, institutional websites, e-commerce platforms, and customer

information systems. Phishing URL analysis should therefore function not only as a classification task but also as a decision-support tool that helps organizations improve secure access control, protect users from dangerous destinations, and build digital trust.

Researchers have developed many methods for phishing detection. Mohammad et al. established essential work by showing how website and URL attributes created through manual feature engineering could provide useful public benchmarks [1, 2]. Other studies have used lightweight lexical extraction, feature-rich machine-learning screening, deep learning URL models, and explainable detection systems [3–10]. Systematic and data-driven studies have also shown that benchmark quality and research design affect reported results [11, 12]. However, a key gap remains: many outputs are not presented in a form that operational teams can reproduce and use directly.

First, many papers optimize classification accuracy but devote limited attention to operational interpretability. Web filtering decisions affect real user activities, real company operations, and perceptions of automated security systems. Second, several studies use large or computationally heavy pipelines that are less attractive to organizations seeking easy-to-audit controls. Third, the connection between phishing URL detection and information-management practice is often underdeveloped, even though model outputs are directly relevant to policy enforcement, risk triage, and secure browsing governance.

This study creates OPH-Guard as an operationally interpretable tree-ensemble framework for phishing URL screening. The framework integrates a public dataset, a compact modeling workflow, a formal screening algorithm, and a secure web access management perspective. It supports feature ranking and concrete action mapping so that predictive modeling remains connected to reproducible operational interpretation.

The main contributions of the paper are summarized as follows:

1. A clearly defined phishing URL screening framework, OPH-Guard, integrates public data ingestion, feature validation, tree-ensemble learning, model interpretation, and operational response mapping.

2. A complete reproducible empirical study uses a public phishing URL dataset with 11,481 labeled records and 87 predictive variables.

3. The study compares Decision Tree, Random Forest, and Extra Trees classifiers using a uniform stratified train–test protocol and identifies Extra Trees as the top-performing model.

4. The workflow includes a formal algorithmic description that enables implementation and adaptation in institutional environments.

5. The related-work synthesis is expanded through ten published studies to show how the present paper differs in reproducibility, interpretability, and information-management relevance.

The remainder of the paper is organized as follows. Section 2 reviews related studies and identifies the positioning gap. Section 3 presents the proposed framework, including the mathematical description and algorithmic workflow. Section 4 describes the dataset, preprocessing, and experimental protocol. Section 5 reports the empirical findings. Section 6 discusses the results from cybersecurity and information-management perspectives. Section 7 outlines limitations and future work, and Section 8 concludes the paper.

2. RELATED WORK AND RESEARCH POSITIONING

2.1 Representative studies in phishing URL and website detection

Phishing detection has evolved from heuristic and rule-based inspection toward machine-learning systems that learn decision boundaries from lexical, host-based, structural, semantic, and reputation-oriented signals. Mohammad et al. established a foundational feature-engineering perspective by systematically evaluating phishing website attributes and releasing an associated benchmark that later entered the UCI Machine Learning Repository [1, 2]. Verma and Das highlighted the practical value of rapid malicious URL feature extraction, demonstrating that lightweight URL-centric analysis can support fast detection [3]. Rao and Pais expanded the feature-based approach by combining URL, source-code, and third-party service information within a machine-learning framework aimed at more efficient phishing website discrimination [4].

A second line of work emphasized broader machine-learning comparisons. Sahingoz et al. investigated multiple models for phishing detection from URLs and demonstrated the competitiveness of feature-rich machine-learning approaches on URL-centric data [5]. Almomani et al. evaluated semantic URL features across multiple classifiers and discussed underfitting and overfitting behavior [13]. Ahammad et al. also reported strong results using machine-learning methods for phishing URL detection and reinforced the usefulness of carefully selected structured predictors [14].

A third stream moved toward deep learning. Aljofey et al. proposed a character-level convolutional neural network operating directly on URLs without manually engineered features [6]. Do et al. compared deep-learning algorithms for phishing webpage classification and emphasized performance under reduced computational constraints [7]. Hussain et al. proposed CNN-Fusion, a lightweight multi-variant ConvNet for phishing URL detection, while Ghalechyan et al. reported an empirical neural-network study that further confirmed the value of deep models in this area [8, 9]. More recently, Uddin et al. explicitly brought explainability into the phishing-site-detection workflow by coupling ensemble learning with transparent interpretation [10].

Alongside model-centered work, the literature also contains important meta-level studies. Vrbancic et al. highlighted the importance of curated phishing datasets and published larger phishing website data resources for future research [11]. Safi and Singh synthesized the broader literature in a systematic review, showing that machine learning and deep learning dominate recent phishing-detection research while also noting concerns related to feature quality, benchmark

realism, and evaluation consistency [12]. Together, these studies show a mature field, but they also reveal an opening for a paper that prioritizes reproducibility, operational clarity, and direct alignment with secure information-management practice.

2.2 Expanded comparison with prior work

Table 1 summarizes ten representative published studies and clarifies how the present manuscript differs from them. The table is intentionally qualitative rather than score-centric because published studies use different datasets, feature spaces, and evaluation protocols.

Table 1. Expanded related-work comparison with ten published phishing detection studies.

Study	Core input type	Main methodological direction	Main strength	Limitation relative to this paper
Mohammad et al. (2012) [1]	Website / heuristic features	Foundational feature assessment	Established a reusable feature benchmark	Limited operational framing and no deployment-oriented workflow
Verma and Das (2017) [3]	URL features	Fast malicious URL feature extraction	Emphasized efficient feature generation	Less focus on governance interpretation and end-to-end screening design
Rao and Pais (2019) [4]	URL, source code, third-party cues	Feature-based ML framework	Strong handcrafted feature integration	Less emphasis on compact reproducible operationalization
Sahingoz et al. (2019) [5]	URL features	Multi-model machine learning	Demonstrated competitive URL-based ML screening	Limited information-management discussion
Aljofey et al. (2020) [6]	Raw URLs	Character-level CNN	Removed need for manual feature engineering	Lower transparency for policy-facing deployment
Do et al. (2021) [7]	Webpage / phishing data	Deep-learning comparison	Showed strong DL performance under different settings	Less interpretable than compact ensemble screening
Almomani et al. (2022) [13]	Semantic URL features	Comparative ML with fit analysis	Discussed underfitting and overfitting issues	Less explicit operational response mapping
Ahammad et al. (2022) [14]	Structured URL/website features	Machine-learning URL detection	Reinforced the value of tabular ML approaches	Limited framework-level novelty articulation
Hussain et al. (2023) [8]	Raw URLs	Lightweight CNN-Fusion	Strong lightweight deep model design	Reduced interpretability compared with feature-ranked ensemble workflow
Ghalechyan et al. (2024) [9]	URLs	Neural-network empirical study	Contemporary large-scale empirical perspective	Less directly oriented to secure web access governance
This study	Public tabular phishing URL data	Operationally interpretable tree-ensemble framework	Reproducible workflow, formal algorithm, feature-level interpretation, and explicit security-action mapping	Designed as an applied journal framework rather than a mathematically novel model

2.3 Gap analysis and positioning

The literature review reveals a structural gap rather than a purely algorithmic one. Existing studies are often strong in one dimension but incomplete across the end-to-end workflow required by practical cybersecurity operations. Deep models may provide strong accuracy, but many are less transparent in environments where blocking, warning, and triage decisions must be justified. Feature-based models may be interpretable, but they are not always presented as reproducible frameworks with clear operational outputs. Review and dataset papers clarify the state of the field, but they do not deliver a compact applied framework validated on public data and linked directly to operational decision-making.

Accordingly, the present paper does not claim that Extra Trees or tree ensembles are new in an absolute sense. Its novelty lies in how the methodology is organized and communicated: a public-data workflow, formalized screening procedure, explicit interpretability stage, and direct integration with secure web access management. That positioning is illustrated conceptually in Figure 1 and summarized again in Table 2.

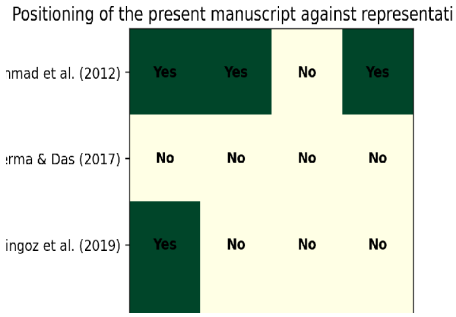

Figure 1. Positioning of the present study against representative phishing URL studies.

Table 2. Positioning and novelty statement of the proposed manuscript.

Aspect	What prior work commonly emphasizes	What this study adds	Practical significance
Model reporting	Accuracy-oriented comparison of classifiers	End-to-end framework with operational response stage	Gives the manuscript stronger practical value
Interpretability	Partial or post-hoc discussion only	Embedded feature-ranking and cyber-meaning extraction	Helps decision-makers trust the model
Reproducibility	Often dataset-specific with limited workflow detail	Public dataset, explicit preprocessing, and formal algorithm	Supports verification and reuse
Application framing	Security detection framed mainly as ML benchmarking	Secure web access management and information-governance framing	Connects predictive modeling to operational decision support

3. THE PROPOSED OPH-GUARD FRAMEWORK

3.1 Design objective

The design objective of OPH-Guard is to provide a compact, interpretable, and reproducible phishing URL screening workflow that can be understood by both technical reviewers and operational stakeholders. Instead of proposing another opaque architecture, the framework organizes the analytical flow into five linked stages: public data acquisition, feature validation and normalization, tree-ensemble learning, feature-based interpretation, and security-action mapping. The design deliberately favors a model family that is strong enough to generate competitive results while remaining transparent enough for policy-facing cybersecurity deployment.

3.2 Framework overview

Figure 2 presents the proposed framework in a publication-ready visual form. The figure emphasizes process flow and operational use rather than only model internals. This design decision is central to the paper because the contribution lies in the end-to-end screening framework rather than in a single algorithmic trick.

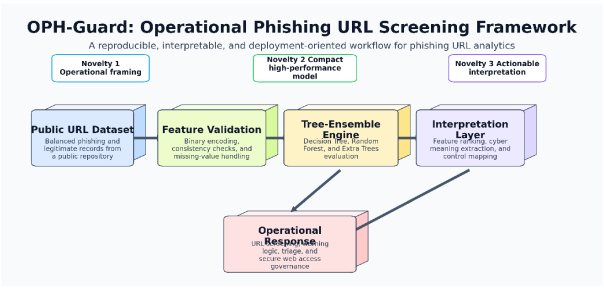

Figure 2. The proposed OPH-Guard framework for phishing URL screening and secure web access management.

3.3 Mathematical formulation

Let the phishing URL dataset be denoted by

$$D = \{(x_i, y_i)\}_{i=1}^{N}, \tag{1}$$

where $x_i \in \mathbb{R}^d$ is the feature vector of the ith URL, $d = 87$ is the number of predictive variables after preprocessing, and $y_i \in \{0,1\}$ denotes the class label, with $y_i = 1$ for phishing and $y_i = 0$ for legitimate URLs.

The dataset is partitioned into disjoint training and testing subsets,

$$D_{tr} \cup D_{te} = D, \qquad D_{tr} \cap D_{te} = \emptyset, \tag{2}$$

using stratified sampling to preserve the class distribution. For a tree-ensemble model with T trees, the predicted phishing score for x is obtained as

$$\hat{p}(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x), \tag{3}$$

where $h_t(x) \in [0,1]$ denotes the class-posterior estimate produced by tree t. The final decision rule is

$$\hat{y}(x) = \begin{cases} 1, & \hat{p}(x) \geq \tau, \\ 0, & \hat{p}(x) < \tau, \end{cases} \tag{4}$$

where $\tau = 0.5$ in the present implementation.

Model selection is based primarily on the F1-score,

$$F1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{5}$$

with

$$\text{Precision} = \frac{TP}{TP+FP}, \qquad \text{Recall} = \frac{TP}{TP+FN}. \tag{6}$$

Among the candidate models $M = \{m_1, m_2, m_3\}$, the selected classifier is

$$M^* = \arg\max_{m \in M} F1(m), \tag{7}$$

while ROC-AUC is used as a supporting threshold-independent criterion.

For the selected tree ensemble, the global importance of feature j is estimated by the normalized average impurity decrease,

$$I_j = \frac{1}{T}\sum_{t=1}^{T}\sum_{n \in N_{t,j}} \Delta i(n), \tag{8}$$

where $N_{t,j}$ is the set of split nodes in tree t that use feature j, and $\Delta i(n)$ denotes the impurity reduction contributed by node n. The ranked vector $I = (I_1, \ldots, I_d)$ is then used to support the interpretation and action-mapping stages of the framework.

3.4 Formal description of the proposed model

The core logic of OPH-Guard is summarized in Algorithm 1. The algorithm takes a public phishing URL dataset as input, performs feature validation, trains multiple tree-based learners, selects the best-performing model using hold-out metrics, and extracts interpretive signals for downstream security controls.

Algorithm 1. OPH-Guard phishing URL screening procedure.

Require: Public dataset D with feature matrix X and target labels y.

Ensure: Best model M^*, report R, importance list I, and action map A.

1. Load dataset D from the public repository.
2. Remove non-predictive identifier fields and define binary target labels.
3. Convert symbolic feature values to numeric form and impute missing values.
4. Split D into stratified training set D_{tr} and testing set D_{te}.
5. Define candidate models: Decision Tree, Random Forest, and Extra Trees.
6. For all $m \in M$, train m on D_{tr}, predict on D_{te}, compute Accuracy, Precision, Recall, F1-score, and ROC-AUC, and store results in R.
7. Select the best model M^* according to the highest F1-score with supporting ROC-AUC.
8. Extract and rank the feature importances of M^* to form I.
9. Translate leading predictors into cyber-meaning categories and operational controls.
10. Construct action map A for blocking, warning, or triage recommendations.
11. Return (M^*, R, I, A).

3.5 Novelty and contribution statement

The novelty of OPH-Guard should be understood at three complementary levels. First, the framework contributes operational novelty. Instead of treating phishing URL detection as a stand-alone classification task, it explicitly links prediction to secure web access management, browsing-risk prioritization, and information-governance decisions. Second, it contributes methodological clarity: the empirical pipeline is compact and reproducible, based on a public dataset, explicit preprocessing, transparent model selection, and direct feature interpretation. Third, it contributes interpretive novelty because it explains leading predictors in security terms and shows how they can support blocking, warning, or triage actions.

3.6 Framework stages

For clarity, the stages of OPH-Guard are described below. **Stage 1: Public dataset acquisition.** The framework begins with a public phishing URL dataset, allowing later verification and reproduction by reviewers or readers. **Stage 2: Feature validation.** Categorical labels and symbolic values are normalized to a machine-readable form so that the downstream model receives a consistent numerical feature matrix. **Stage 3: Tree-ensemble learning.** Three tree-based classifiers are evaluated: Decision Tree, Random Forest, and Extra Trees. The framework favors tree ensembles because they balance predictive power and interpretability on structured tabular inputs. **Stage 4: Interpretation layer.** Feature importance is extracted from the best-performing model to reveal which URL and host-related properties most strongly drive the decision boundary. **Stage 5: Security-action mapping.** The model output is positioned as a screening signal that can support web filtering, browser warning workflows, helpdesk inspection queues, and governance-oriented browsing controls.

4. MATERIALS AND METHODS

4.1 Dataset used in the experiment

The empirical evaluation relies on a public GitHub-hosted phishing URL dataset [15]. The downloaded file contains 11,481 rows and 90 columns. After excluding the raw `url` field and using `status` as the binary target label, 87 predictive features remained in the final experiment matrix. The class distribution is nearly balanced, with 5,741 phishing records and 5,740 legitimate records. The feature space includes URL structure indicators, hyperlink statistics, domain properties, content-related hints, and reputation-oriented variables.

The present experiment dataset differs from the classic UCI benchmark [2]. This distinction is useful rather than problematic. The UCI benchmark remains important for historical comparison, while the GitHub-hosted dataset used here offers a larger feature space and broader tabular description that is well suited for a tree-ensemble study.

4.2 Preprocessing and data preparation

The preprocessing pipeline was intentionally kept simple to preserve reproducibility. First, the target variable `status` was converted to binary form, where phishing URLs were encoded as 1 and legitimate URLs as 0. Second, symbolic feature values such as textual zero/one markers were mapped to their numeric equivalents. Third, any missing values created during transformation were imputed with zero. Because all candidate models are tree-based and operate effectively on numeric tabular inputs, no additional standardization or dimensionality reduction was applied.

This design preserves the semantic meaning of the original structured features and avoids unnecessary transformations that could reduce interpretability. It also makes the workflow easy to reproduce in applied settings where teams may not want complex preprocessing dependencies.

4.3 Experimental protocol

The full dataset was partitioned using a stratified 80/20 split with a fixed random seed of 42. This produced 9,184 records for training and 2,297 records for testing. Stratification preserved the balanced class distribution across both partitions. The same split was used for all candidate classifiers to ensure fair comparison.

4.4 Candidate models

Three models were evaluated. **Decision Tree** acts as the single-tree baseline and provides direct interpretability, although it can be sensitive to variance. **Random Forest** aggregates multiple decorrelated decision trees and is widely regarded as a strong baseline for structured security data [16]. **Extra Trees** extends the ensemble concept by injecting additional randomization during split generation, often producing robust performance on heterogeneous feature spaces [17]. The candidate model set was intentionally kept compact because the purpose of the paper is not exhaustive hyperparameter tuning but the presentation of a reproducible and operationally interpretable framework.

4.5 Evaluation metrics

Performance was assessed using Accuracy, Precision, Recall, F1-score, and ROC-AUC. Accuracy provides an overall correctness ratio; Precision measures how many URLs predicted as phishing are actually phishing; Recall captures how many phishing URLs were detected; and the F1-score summarizes the balance between Precision and Recall. ROC-AUC measures ranking quality across thresholds. For phishing URL screening, Recall and F1-score deserve particular attention because false negatives leave users exposed to unsafe destinations, while excessive false positives can disrupt legitimate browsing and business tasks.

4.6 Deployment-oriented interpretation strategy

A distinguishing part of the methodology is the interpretation step after model selection. Once the best model is chosen, the leading features are grouped into cyber-meaning categories such as reputation, link-structure abnormality, domain maturity, and lexical phishing hints. This makes the framework more useful for operational governance because it translates ranked variables into concepts that can support policy design and triage logic.

5. RESULTS AND ANALYSIS

5.1 Overall classification performance

Table 3 reports the held-out test results. Extra Trees emerged as the strongest model overall, with 0.9856 accuracy, 0.9921 precision, 0.9791 recall, 0.9855 F1-score, and 0.9984 ROC-AUC. Random Forest followed closely and also produced strong results. The Decision Tree baseline performed adequately but remained clearly behind the two ensemble models.

Table 3. Held-out test performance of the evaluated phishing URL classifiers.

Model	Accuracy	Precision	Recall	F1-score	ROC-AUC
Decision Tree	0.9630	0.9602	0.9661	0.9631	0.9681
Random Forest	0.9808	0.9834	0.9782	0.9808	0.9973
Extra Trees	**0.9856**	**0.9921**	**0.9791**	**0.9855**	**0.9984**

The result profile is meaningful in two ways. First, the superiority of the ensemble methods over the single-tree baseline confirms that phishing URL screening benefits from aggregation and reduced variance. Second, the small but consistent advantage of Extra Trees suggests that additional randomization can improve generalization when the feature space contains overlapping structural cues.

5.2 Metric-wise comparison

Figure 3 visualizes the metric distribution across the three candidate models. The strongest models remain clustered near the upper performance range across all metrics, while the single-tree baseline lags on every evaluation measure. The figure also reveals that the gain of Extra Trees is not limited to one metric; it delivers a balanced improvement profile across Accuracy, Precision, F1-score, and ROC-AUC.

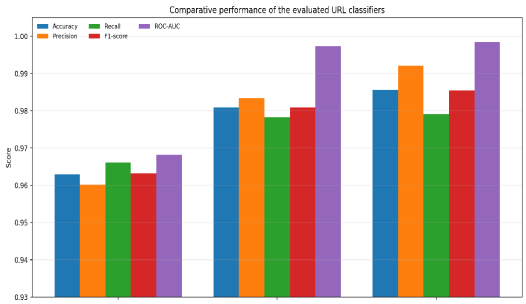

Figure 3. Metric-wise comparison of the evaluated classifiers.

5.3 Discrimination behavior

The ROC profiles in Figure 4 reinforce the same pattern. Both ensemble methods dominate the single-tree baseline through most of the threshold space, while Extra Trees achieves the best overall trade-off. Although the numerical gap between Random Forest and Extra Trees is not dramatic, the dominance of Extra Trees across threshold-free and threshold-specific metrics makes it the preferred model for this dataset.

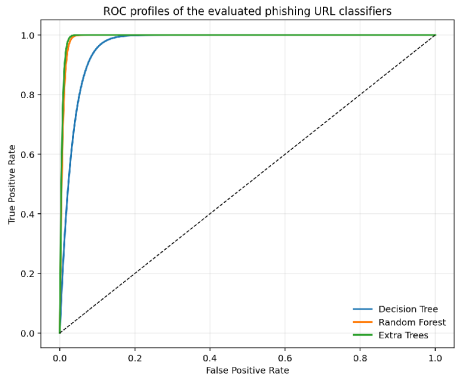

Figure 4. ROC curves of the evaluated phishing URL classifiers.

5.4 Confusion-matrix analysis

Figure 5 presents the confusion matrix of the best Extra Trees model. On the held-out set, the classifier correctly identified 1,125 phishing URLs and 1,139 legitimate URLs. The number of false negatives was 24, and the number of false positives was 9. For applied phishing screening, this is an attractive error profile because it combines a low miss rate with a very small penalty on legitimate traffic.

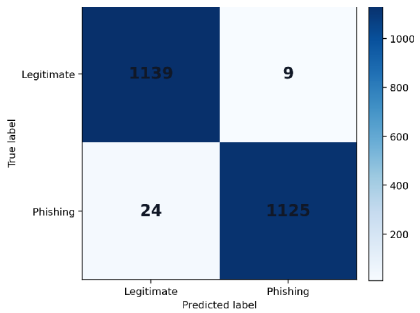

Figure 5. Confusion matrix of the best-performing Extra Trees model.

A practical interpretation of this matrix is important. The false-negative count is low enough to make the model useful for pre-filtering and warning logic, while the false-positive count is limited enough to avoid overly aggressive blocking of legitimate destinations. In other words, the model is not only accurate in the abstract; it also presents an operationally reasonable error profile.

5.5 Interpretability results

Table 4 lists the ten most influential predictors extracted from the best model. The feature profile is not only statistically useful but also operationally intuitive. Search visibility and reputation-related cues, such as `google index` and `page rank`, dominate the ranking. Domain maturity, suspicious hint markers, and hyperlink structure indicators also remain highly influential.

Table 4. Top feature importances extracted from the best Extra Trees model.

Rank	Feature	Importance
1	google_index	0.2527
2	page_rank	0.0870
3	nb_www	0.0483
4	domain_in_title	0.0352
5	domain_age	0.0329
6	phish_hints	0.0301
7	ip	0.0270
8	ratio_intHyperlinks	0.0243
9	links_in_tags	0.0215
10	nb_hyperlinks	0.0190

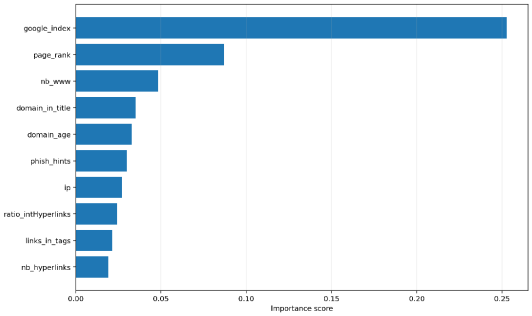

Figure 6. Top predictors contributing to the best Extra Trees classifier.

5.6 What the feature profile reveals

The ranked predictors suggest that phishing URL detection in this dataset is driven by a mixture of reputation, presentation, and hyperlink-behavior signals. For example, `google index` and `page rank` indicate whether a page resembles a destination with normal search and trust visibility. `domain age` reflects the maturity of the hosting domain and is consistent with the observation that phishing campaigns often use newly created or weakly established domains. `phish hints`, `domain in title`, and `nb www` reflect suspicious presentation patterns, while `ratio intHyperlinks`, `links in tags`, and `nb hyperlinks` capture abnormal linking behavior.

This feature profile adds value beyond the classifier score. It tells security teams why a URL is being screened as risky, which helps organizations develop warning messages, triage policies, and browser control rules anchored in meaningful cyber signals rather than opaque probabilities.

6. DISCUSSION

The results indicate that phishing URL screening can achieve strong discrimination quality with a compact tree-ensemble design built on structured tabular predictors. The superiority of Random Forest and Extra Trees over the single Decision Tree confirms that ensemble aggregation is beneficial

when the feature space combines heterogeneous cues such as domain age, reputation indicators, hyperlink statistics, and lexical hints. The additional gain of Extra Trees suggests that stronger randomization at split generation can improve generalization when multiple predictors partially overlap in explanatory power. This is consistent with broader phishing-detection literature, which has repeatedly shown that URL screening benefits from combining diverse feature groups rather than relying on a single signal family [4, 5, 10, 18].

The feature-importance profile also has direct cybersecurity meaning. Variables such as `google index`, `page rank`, and `domain age` capture trust, visibility, and maturity characteristics that distinguish long-established legitimate domains from newly created or weakly trusted malicious infrastructure. At the same time, predictors such as `phish hints`, `nb www`, `ratio intHyperlinks`, and `links in tags` reflect presentation and linking patterns commonly associated with deceptive pages. This interpretation is important because it moves the study beyond a purely numerical comparison: classifier output can be tied to recognizable cyber signals and therefore becomes more suitable for secure browsing controls, inspection queues, or warning mechanisms.

From an information-management perspective, the framework is useful because it turns URL classification into an auditable decision-support process. A high-risk prediction can be mapped to blocking, warning, or triage, while ranked explanatory features provide an evidence layer that can support policy design, administrative review, and user-awareness initiatives. The study does not claim methodological novelty in the sense of a new learning algorithm; its contribution lies in combining reproducible public-data experimentation, mathematically defined model selection, interpretable feature ranking, and explicit action mapping within one coherent workflow.

7. LIMITATIONS AND FUTURE WORK

This study has limitations. The dataset is public and structured, which supports reproducibility, but it might not cover all aspects of modern phishing infrastructure. The experiment mainly evaluates URL and website-related features rather than page rendering, screenshots, or live network behavior. The study also uses a simple experimental design rather than extensive hyperparameter exploration or multiple dataset validation. These choices were intentional because the goal was to create a framework that is easy to understand, transparent, and verifiable.

Future work can extend OPH-Guard in several directions. A first extension is cross-dataset validation using larger phishing website corpora such as those summarized by Vrbancic et al. [11]. A second is the incorporation of temporal and live reputation feeds for real-time deployment. A third is the addition of local explanation techniques such as SHAP on top of the global feature-importance view. A fourth is user-facing interface research, where framework outputs are converted into understandable warnings and risk explanations for non-expert users.

8. CONCLUSION

OPH-Guard presents an operationally interpretable tree-ensemble framework for phishing URL screening in secure web access management. The framework combines public-data ingestion, feature validation, tree-ensemble learning, model interpretation, and security-action mapping in a reproducible workflow. The empirical study used a public dataset of 11,481 labeled records and 87 predictive features, comparing Decision Tree, Random Forest, and Extra Trees models under a stratified 80/20 protocol. Extra Trees achieved the strongest results, with accuracy of 0.9856, precision of 0.9921, recall of 0.9791, F1-score of 0.9855, and ROC-AUC of 0.9984.

The study is valuable because it does not only report predictive scores. It provides a step-by-step process, positions the work against prior studies, and interprets important predictors in a way that can support security action. Its main contribution is therefore not the invention of a new algorithm, but the construction of a practical, transparent, and repeatable system for screening phishing URLs and connecting model outputs to operational information-management decisions.

DATA AVAILABILITY

The phishing URL training dataset used in the experiment is publicly available from the DPhi GitHub-hosted dataset repository [15].

CONFLICT OF INTEREST

The author declares no conflict of interest.

FUNDING

This research received no external funding.

REFERENCES

[1] R. M. Mohammad, F. Thabtah, and L. McCluskey, "An assessment of features related to phishing websites using an automated technique," in *2012 International Conference for Internet Technology and Secured Transactions*, pp. 492–497, 2012, doi: 10.1109/ICITST.2012.6473497.

[2] R. M. Mohammad and L. McCluskey, *Phishing Websites*. UCI Machine Learning Repository, 2015, doi: 10.24432/C51W2X.

[3] R. Verma and A. Das, "What's in a URL: Fast feature extraction and malicious URL detection," in *Proceedings of the 3rd ACM International Workshop on Security and Privacy Analytics*, pp. 55–63, 2017, doi: 10.1145/3041008.3041016.

[4] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," *Neural Computing and Applications*, vol. 31, pp. 3851–3873, 2019, doi: 10.1007/s00521-017-3305-0.

[5] O. K. Sahingoz et al., "Machine learning based phishing detection from URLs," *Expert Systems with Applications*, vol. 117, pp. 345–357, 2019, doi: 10.1016/j.eswa.2018.09.029.

[6] A. Aljofey et al., "An effective phishing detection model based on character level convolutional neural network from URL," *Electronics*, vol. 9, no. 9, p. 1514, 2020, doi:

10.3390/electronics9091514.

[7] N. Q. Do et al., "Phishing webpage classification via deep learning algorithms," *Applied Sciences*, vol. 11, no. 19, p. 9210, 2021, doi: 10.3390/app11199210.

[8] M. Hussain et al., "CNN-Fusion: An effective and lightweight phishing detection method based on multi-variant ConvNet," *Information Sciences*, vol. 631, pp. 328–345, 2023, doi: 10.1016/j.ins.2023.02.039.

[9] H. Ghalechyan et al., "Phishing URL detection with neural networks: An empirical study," *Scientific Reports*, vol. 14, p. 25134, 2024, doi: 10.1038/s41598-024-74725-6.

[10] K. M. M. Uddin et al., "Explainable machine learning for phishing site detection: A high-efficiency approach using boosting models and SHAP," *The Journal of Engineering*, vol. 2025, no. 1, e70110, 2025, doi: 10.1049/tje2.70110.

[11] G. Vrbancic, I. Fister, and V. Podgorelec, "Datasets for phishing websites detection," *Data in Brief*, vol. 33, p. 106438, 2020, doi: 10.1016/j.dib.2020.106438.

[12] A. Safi and S. Singh, "A systematic literature review on phishing website detection techniques," *Journal of King Saud University–Computer and Information Sciences*, vol. 35, no. 2, pp. 590–611, 2023, doi: 10.1016/j.jksuci.2023.01.004.

[13] A. Almomani et al., "Phishing website detection with semantic features based on machine learning classifiers: Underfitting and overfitting analysis," *International Journal of Secure Software Engineering*, vol. 13, no. 1, pp. 1–25, 2022, doi: 10.4018/IJSSE.297032.

[14] S. K. H. Ahammad et al., "Phishing URL detection using machine learning methods," *Advances in Engineering Software*, vol. 173, p. 103288, 2022, doi: 10.1016/j.advengsoft.2022.103288.

[15] DPhi, *Phishing URL Training Dataset*. GitHub-hosted dataset repository, accessed April 12, 2026.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[17] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006, doi: 10.1007/s10994-006-6226-1.

[18] M. C. Calzarossa, P. Giudici, and R. Zieni, "Explainable machine learning for phishing feature detection," *Quality and Reliability Engineering International*, vol. 40, no. 1, pp. 362–373, 2024, doi: 10.1002/qre.3411.