



Reduced Stacking Classifier Model with Optimized feature Selection for Efficient Phishing Website Detection

¹I. N. Thompson, ²V. T. Emmah & ³D. Mathias

^{1,2&3}Department of Computer Science, Rivers State University, Port-Harcourt, Nigeria

Email: ¹thompson.iyelisi@rsu.edu.ng, ²victor.emmah@ust.edu.ng, ³daniel.mathias@ust.edu.ng

Abstract

Phishing websites remain an effective vehicle for credential theft and financial fraud, while machine-learning detectors must increasingly balance predictive performance against the cost of deployment. Stacking ensembles can improve classification by combining diverse learners, but large ensembles operating on full feature sets increase inference cost. This paper examines an HHO–GWO reduced stacking model for phishing website classification. It uses Harris Hawks Optimisation (HHO) for feature selection, Random Forest, XGBoost, and Gradient Boosting classifiers, and Grey Wolf Optimiser (GWO) weights for combining classifier probabilities. The model was evaluated on the UCI Phishing Websites dataset and the Mendeley Web Page Phishing Detection dataset using five random seeds, and compared with individual classifiers, unoptimised stacking, a replicated nine-classifier stacking baseline, and ablated configurations. It obtained mean accuracies of 94.40% and 96.38% and mean F1-scores of 94.53% and 96.31% on UCI and Mendeley, respectively. It used an average of 23.2 of 30 UCI features and 65.0 of 87 Mendeley features. Relative to the nine-classifier baseline, mean inference latency declined from 1.924 ms to 0.465 ms on UCI and from 1.457 ms to 0.271 ms on Mendeley, corresponding to reductions of 75.8% and 81.4%. The associated accuracy differences were 0.47 and 0.39 percentage points. The findings position HHO–GWO reduced stacking as a practical accuracy–efficiency alternative for offline phishing classification experiments. Its suitability for live URL screening or evolving attack streams requires further validation.

Keywords:

Phishing Stacking ensemble; Harris Hawks Optimisation; Grey Wolf Optimiser; feature selection.

1. Introduction

Phishing attacks use deceptive websites, messages, and domains to induce users to disclose credentials, financial information, or other sensitive data. Website and URL classification is an important defensive point of intervention because it can identify suspicious destinations before a user submits information to a fraudulent page. However, the structural and content-derived signals available to a detector are diverse and imperfect: URL lexical characteristics, domain properties, hyperlinks, page behaviour, and reputation-related indicators may each contribute evidence, yet no individual signal is sufficient on its own. Machine-learning methods are

therefore widely used to combine such evidence into a phishing or legitimate classification [1], [2].

The practical challenge is not classification quality alone. A detector intended for a latency-sensitive workflow must also control the number of features processed and the computational cost of its decision procedure. Conventional single classifiers, including Random Forest and gradient-boosting variants, can produce strong results on benchmark phishing datasets, but their performance varies with the feature representation and data distribution [2], [3]. Ensemble learning can exploit complementary error patterns among heterogeneous models. In particular, stacking trains base classifiers and combines their outputs in a second-stage decision layer, which can improve robustness when the constituent models have useful diversity [4].

That benefit may come with considerable inference overhead. Large stacking configurations require every base learner to generate a prediction for each input, and the cost is amplified when all available features are passed through each learner. A published phishing-detection stack, for example, used nine base classifiers [5]. Recent phishing studies also report high-performing stacked architectures, feature-selection pipelines, and optimisation mechanisms, but their results often emphasise peak predictive scores rather than the trade-off among feature count, number of classifiers, and per-instance latency [6]–[8]. This distinction matters when a classifier is assessed as a component that may later support a browser, endpoint, or gateway workflow, even when the present experiment is conducted offline.

Feature selection offers one route to reducing this cost. Meta-heuristic search is attractive for phishing data because it can search combinations of heterogeneous indicators without assuming that their contribution is independent or linear. Harris Hawks Optimisation (HHO) is a population-based optimisation method that alternates exploration and exploitation while searching for a high-quality solution [9]. In a feature-selection setting, a candidate solution represents a subset of the available features, permitting the search to balance predictive utility with parsimony. HHO and other nature-inspired strategies have been applied to phishing and cyber-security classification problems, although reduction in input dimensionality does not by itself resolve the cost of a large ensemble [3], [7].

The combination layer of a stack also affects both its behaviour and interpretability. A conventional meta-learner commonly uses fixed fitted coefficients for base-model outputs. An alternative is to optimise a non-negative, normalised vector of base-classifier weights against a validation objective. The Grey Wolf Optimiser (GWO) is a swarm-based search method in which candidate solutions are updated with reference to leading solutions in the population [10]. Its low-dimensional use for three probability weights makes it feasible to search for a dataset- and split-specific combination without adding another learned classifier to the inference pipeline. Such optimisation should not be interpreted as evidence of temporal adaptation to new phishing campaigns; it only selects weights using the validation data available in a given experiment.

This paper investigates whether feature selection and reduced weighted stacking can jointly provide a favourable accuracy–efficiency balance for phishing website classification. The HHO–GWO model applies HHO to select a subset of numerical phishing indicators, trains Random Forest, XGBoost, and Gradient Boosting base classifiers on the selected subset, and applies GWO to determine their normalised probability-combination weights. Evaluation uses

two public benchmark datasets with different feature dimensionalities: the UCI Phishing Websites dataset and the Mendeley Web Page Phishing Detection dataset [11], [12]. Experiments use five random seeds and compare the HHO–GWO model with individual classifiers, conventional three-classifier stacking, a replicated nine-classifier stacking baseline [5], and two ablated versions.

This paper details a reproducible reduced stacking design that integrates HHO feature selection with GWO probability weighting using three heterogeneous tree-based classifiers. It evaluates the design on two benchmark datasets using repeated experiments and comparators that separate the roles of feature selection, probability weighting, classifier count, and conventional stacking. It also quantifies the resulting accuracy–efficiency trade-off: .

2. Related Works

This section reviews studies that have investigated stacking ensemble architectures, meta-heuristic optimisation algorithms, and their intersection in the domain of phishing and cyberattack detection. Each study is examined with attention to its research objective, methodology, key findings, and limitations. The review covers 41 papers published between 2024 and 2026, reflecting recent work in optimised ensemble-based phishing detection.

Jain et al. (2026) developed PhishNet 1.0, a phishing URL detection framework that combines Boruta-based heuristic feature selection with an Optuna-optimised stacking ensemble. The base learner layer comprises Gradient Boosting, K-Nearest Neighbours, and Decision Trees, while Optuna’s Tree-structured Parzen Estimator serves as the search strategy for hyperparameter tuning of the meta-learner. The framework achieved 96.15% accuracy on the evaluated dataset, with the authors reporting better performance than traditional grid and random search methods in identifying stacking configurations. The study’s limitation lies in the challenge of maintaining consistent real-time performance when processing rapidly evolving, high-dimensional URL data in production environments.

Wabi et al. (2024) evaluated a stacking ensemble framework by deploying nine base classifiers, namely XGBoost, CatBoost, LightGBM, Random Forest, Gradient Boosting, Extra Trees, SVC, AdaBoost, and Bagging, across three independent phishing datasets. The framework achieved detection accuracies of 97.16%, 98.58%, and 97.39% on the respective datasets, suggesting that stacking diverse learners can support stability across different data distributions. However, the use of nine base classifiers introduces computational overhead that the authors acknowledge may hinder deployment in resource-constrained real-time environments, highlighting the trade-off between ensemble diversity and operational efficiency.

Kanathey et al. (2025) proposed the Optimized Stacking Ensemble Learning Model (OSELN), which integrates nature-inspired meta-heuristic algorithms to refine both base learner parameters and the meta-model layer within a stacking architecture. The meta-heuristic optimisation led to measurable improvements in detection precision and F1-score compared to standard stacking ensembles, with the model showing particular strength against borderline samples that conventional stacking approaches frequently misclassify. The computational expense of the nature-inspired search during the training phase was identified as the primary limitation, requiring careful balancing of search iterations against training time budgets.

Bayraktar (2026) designed a stacking-based phishing website detection system using Logistic Regression, K-Nearest Neighbours, and Naive Bayes as base classifiers. The system reported 99.99% accuracy on the evaluated benchmark dataset. This result is notable, but it also raises questions about potential overfitting to the specific benchmark dataset, and the study does not address feature selection, meta-learner adaptability, or performance on rapidly evolving real-world phishing campaigns.

Upreti et al. (2025) introduced AQUAPHISH, a framework that addresses detection latency through meta-heuristic feature selection using the Whale Optimisation Algorithm. The WOA reduced the feature set from 48 to 36 selected features, and these features were processed through a stacked ensemble generated by H2O AutoML. The framework achieved 98.3% detection accuracy with lower inference latency compared to full-feature-set models, making it more suitable for real-time browser integration. The reliance on AutoML creates a black-box meta-learner that is difficult to interpret or manually tune for specific edge cases in phishing detection.

Hamdan et al. (2024) conducted a comparative study of Grey Wolf Optimiser and Particle Swarm Optimisation for feature selection in phishing classification, evaluating the selected features using an ensemble of classifiers including XGBoost and Random Forest. The study found that GWO navigated complex feature spaces with fewer iterations than standard PSO, though the combined PSO-GWO hybrid approach achieved an F-measure of only 91.4%. This relatively modest performance suggests that feature selection alone, without a sophisticated meta-learner architecture, may be insufficient for achieving competitive detection rates.

Pathak and Shrivastava (2024) applied Particle Swarm Optimisation to determine feature weights within a Random Forest-based classification framework for phishing website detection. The PSO-based feature optimisation improved detection rates compared to models trained on full feature sets, suggesting that meta-heuristic search can identify discriminative patterns in URL-based features that manual engineering or statistical filtering methods may overlook. The study's scope was limited to Random Forest as the primary base learner, leaving the impact of PSO-optimised features on more complex stacking meta-learners unexplored.

Patil et al. (2025) proposed a phishing detection framework that integrates nature-inspired meta-heuristic algorithms with a multi-layered stacking ensemble of diverse base learners. The framework reduced the false positive rate in real-time detection scenarios, with the stacking architecture reporting improved performance against obfuscated phishing links. The computational complexity of the biological meta-heuristic increases processing time when applied to large-scale data streams, a constraint that may limit deployment in high-throughput network monitoring environments.

Subashini and Narmatha (2024) combined deep learning base learners, Bi-LSTM and Faster R-CNN, within a stacking framework and employed the Grey Wolf Optimiser to dynamically tune the weights of the stacking meta-learner. The GWO-tuned model reported better adaptability to zero-day phishing attacks than static stacking configurations, suggesting that dynamic weight optimisation can help the ensemble adjust its reliance on different base classifiers as attack patterns change. The deep learning base learners require substantial training data and computational resources, limiting their applicability in lightweight mobile security deployments.

Gupta et al. (2024) developed a Cuckoo Search-optimised Deep CNN for enhanced cybersecurity in IoT networks. The Cuckoo Search algorithm optimised the hyperparameters of the deep convolutional network, including learning and dropout rates, and the model was benchmarked against stacked ensemble architectures. The framework achieved 91% detection accuracy with reduced computational complexity suitable for IoT-scale processing. The study primarily addressed IoT-scale datasets and may require further scaling for massive enterprise-level URL streams.

Abualhaj et al. (2025) employed Harris Hawks Optimisation for feature selection within a stacking ensemble consisting of XGBoost, Random Forest, and Histogram-Based Gradient Boosting, with an SVM meta-classifier. The HHO-optimised stacking model achieved 99.81% accuracy, and the study reported better convergence speed and feature subset quality than PSO and Genetic Algorithms. The performance on the ISCX-URL2016 dataset, while encouraging, requires cross-dataset validation to assess the generalisability of the HHO-selected feature subset.

Hassan et al. (2024) evaluated stacking configurations against traditional monolithic classifiers for phishing website identification, employing Gradient Boosting, K-Nearest Neighbours, and Support Vector Machines as base learners with a Logistic Regression meta-learner. The stacking model achieved 98% accuracy, and the research suggests that fusing heterogeneous base learners, specifically combining distance-based and tree-based algorithms, can capture more complex data patterns than a single classifier. The study did not address the computational time required for real-time prediction, an important factor for practical browser deployment.

Al-Shalif et al. (2025) introduced the Enhanced Equilibrium Optimizer for feature selection in cyberattack detection, including phishing. The EEO-based stacking model, employing J48 Decision Tree and ExtraTreeClassifier as base learners, achieved 99.7% accuracy while reducing the number of features required for classification. The dimensionality reduction lowered computational load, supporting the argument that meta-heuristic feature selection can improve ensemble efficiency. The primary evaluation was conducted on the NSL-KDD dataset, and performance on modern obfuscated URL datasets remains an area for further investigation.

Bari et al. (2025) proposed a multi-tier stacking ensemble framework integrating filter-based feature selection with the Salp Swarm Algorithm for refinement. The optimised features were processed through a stacked ensemble of Random Forest, AdaBoost, and XGBoost, with the SSA used to fine-tune both feature subsets and base learner parameters. The framework achieved 98.58% accuracy, with the SSA reported to reduce local-optimum problems during the feature search process. The multi-tier selection process adds initial computational complexity during the training phase.

Sheikhi and Kostakos (2024) developed a hybrid framework that combines Firefly Algorithm-based feature selection with PSO-optimised XGBoost for malicious website detection. The Firefly-based feature selection improved detection sensitivity, and the combined FA-PSO-XGBoost approach outperformed traditional ensemble models in identifying obfuscated phishing URLs. The multi-stage optimisation process, employing one meta-heuristic for features and another for model parameters, increases overall training time but shows the viability of applying complementary meta-heuristics at different pipeline stages.

Alshammari et al. (2025) introduced a hybrid phishing detection architecture using the Chaotic Dragonfly Algorithm for automated feature selection. The selected features were evaluated using two stacking configurations: one combining Random Forest, Neural Network, and Bagging, and another combining KNN, Random Forest, and Bagging. The system achieved 97.4% accuracy, and the chaotic enhancement to the dragonfly search reduced the likelihood of convergence to local optima compared to the standard Dragonfly Algorithm. The computational overhead of chaotic mapping and multi-layered stacking requires high-performance hardware for real-time inference.

3. Methodology

The HHO–GWO model combines HHO-driven feature selection with a reduced stacking ensemble and GWO-derived decision weights. The design retains complementary tree-based classifiers while avoiding the feature burden and inference cost of a larger stack. It first reduces the input representation, then combines the probability estimates of a small set of base classifiers through learned non-negative weights.

Figure 1 shows the resulting system architecture. Website-derived numerical URL, domain, webpage, and hyperlink features enter the input layer. The HHO feature-selection component retains the selected feature vector, which is processed in parallel by Random Forest, XGBoost, and Gradient Boosting classifiers. Their phishing probabilities are passed to the GWO-weighted decision-fusion layer, which produces the final phishing probability and classification output. HHO selects the feature subset and GWO determines the fusion weights during model development; the deployed classifier applies the selected features and learned weights to each input vector.

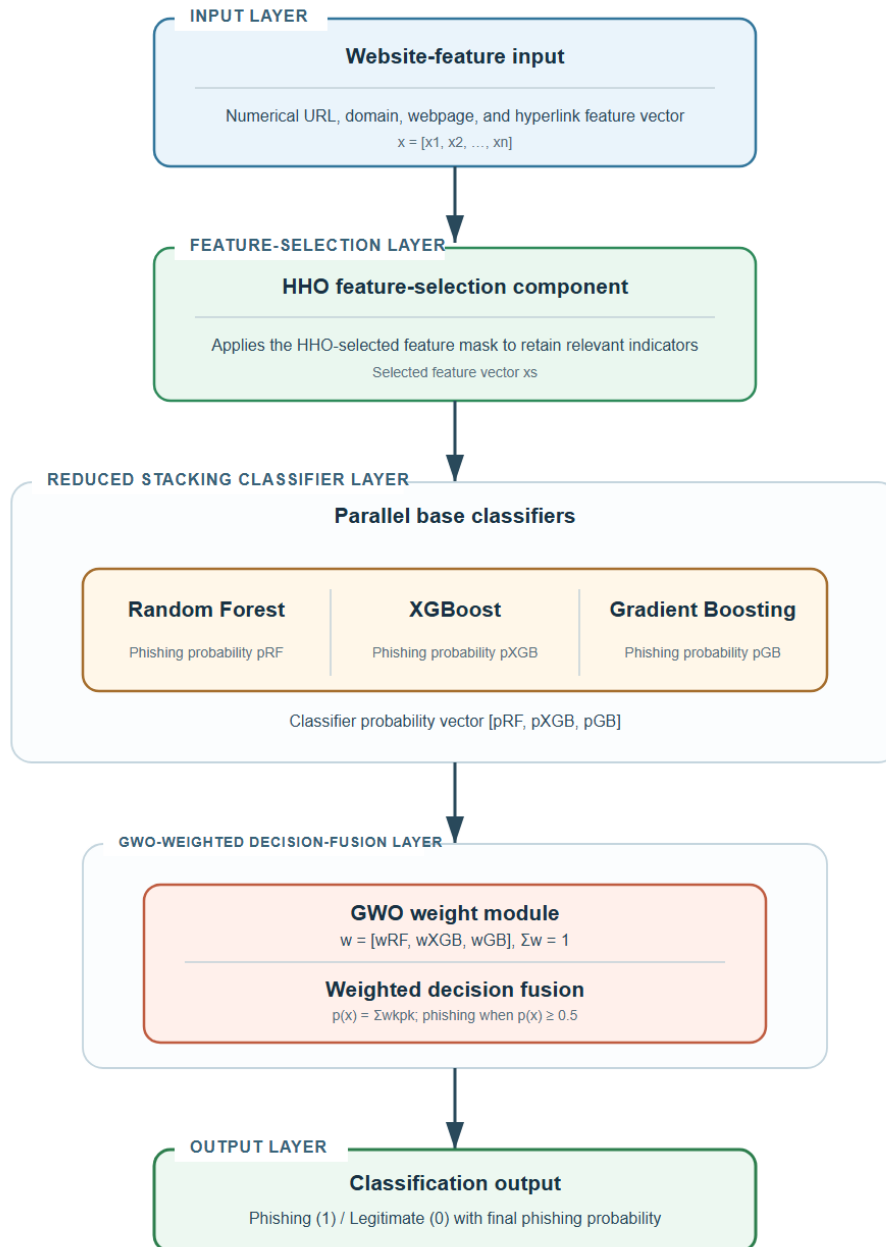


Figure 1: HHO-GWO phishing-detection system architecture.

The base layer contains Random Forest (RF), XGBoost, and Gradient Boosting (GB), giving the ensemble three heterogeneous tree-based probability estimates. Given a selected feature vector x , their estimated phishing probabilities are denoted $p_{RF}(x)$, $p_{XGB}(x)$, and $p_{GB}(x)$. GWO produces a non-negative weight vector $w = (w_{RF}, w_{XGB}, w_{GB})$ constrained so that $\sum_j w_j = 1$. The final phishing probability is

$$p(x) = w_{RF}p_{RF}(x) + w_{XGB}p_{XGB}(x) + w_{GB}p_{GB}(x). \quad (1)$$

Equation (1) computes the final phishing probability from the three base-classifier probabilities. An instance is assigned to the phishing class when $p(x) \geq 0.5$; otherwise, it is assigned to the legitimate class. The weighted rule replaces a fourth learned meta-classifier with validation-optimised, non-negative classifier contributions in the reduced ensemble.

To evaluate this architecture, two public benchmark datasets were used. The UCI Phishing Websites dataset contains 11,055 instances and 30 features describing URL, domain, and page characteristics [11]. The Mendeley Web Page Phishing Detection dataset contains 11,430 instances and, after removal of the URL identifier, 87 usable numerical features covering URL structure, hostname properties, hyperlinks, page content, domain information, and external-service indicators [12]. The UCI labels were harmonised from their original encoding to binary values, while Mendeley text labels were mapped to the same convention: phishing = 1 and legitimate = 0. Table 1 lists the dataset characteristics.

Table 1: Dataset characteristics.

Property	UCI Phishing Websites	Mendeley Web Page Phishing Detection
Raw instances	11,055	11,430
Usable modelling features	30	87
Phishing class	4,898 (44.3%)	5,715 (50.0%)
Legitimate class	6,157 (55.7%)	5,715 (50.0%)
Label used in this study	1 = phishing; 0 = legitimate	1 = phishing; 0 = legitimate

For each dataset, labels were harmonised, duplicate records were removed, and the numerical feature matrix was prepared through median imputation and min–max scaling. The imputation operation was retained as a pipeline safeguard, although no missing values were observed in the usable modelling features. Duplicate removal eliminated 5,206 UCI records and 174 Mendeley records before splitting. The cleaned data were stratified into 70% training, 15% validation, and 15% test partitions. Stratification preserved the relative frequency of phishing and legitimate instances in each partition.

The held-out test partition was not used to select feature masks or GWO weights. Five-fold stratified cross-validation on the training partition supported meta-feature generation and model-related selection steps, while the validation partition supported the optimisation of the probability weights. The complete procedure was repeated for random seeds 42, 0, 1, 2, and 3. Results reported as aggregate performance are means across these five runs.

With these partitions established, HHO represents a candidate feature subset as a binary vector whose elements indicate whether a feature is retained. The search used 30 hawks for up to 100 iterations. Its fitness function prioritised validation classification accuracy while including a secondary penalty for unnecessarily large subsets. The search retained at least 67% of features and required at least 99.5% of full-feature validation accuracy. These constraints retain a near-full-performance feature subset rather than the smallest possible subset.

After selection, RF, XGBoost, and GB were trained on the reduced training representation. RF, XGBoost, and GB each used 100 estimators; the learning rate was 0.1 for XGBoost and GB. Their probability outputs form a three-column meta-feature representation. GWO then initialised 20 candidate weight vectors and updated them for up to 50 iterations. After every update, weights were clipped to [0,1] and normalised to sum to one. Candidate fitness was evaluated against the validation data, and the best final normalised vector was used for the weighted probability rule. A convergence threshold of 10^{-4} allowed the GWO search to stop when improvement became negligible.

The HHO–GWO model was evaluated against five comparison categories:

1. individual RF, XGBoost, and GB classifiers trained with the full feature set;
2. unoptimised three-classifier stacking with a conventional logistic-regression meta-learner and no HHO or GWO;
3. a nine-classifier stacking baseline replicated under the same experimental setting;
4. the complete HHO + three-classifier stacking + GWO framework; and
5. two ablations: stacking + GWO without HHO, and HHO + stacking without GWO.

The nine-classifier baseline was replicated locally under the same experimental conditions; its values are not re-reported published results from [5].

Classification performance was measured with standard accuracy, precision, recall, and F1-score, calculated from true positives, true negatives, false positives, and false negatives.

Mean per-instance inference latency, expressed in milliseconds, was used as the efficiency measure. It reflects the prediction stage and therefore excludes the offline HHO and GWO searches. Feature count was also reported, because it records the dimensionality reduction achieved before classifier inference. The experiments were implemented in Python using pandas, NumPy, scikit-learn, XGBoost, LightGBM, matplotlib, seaborn, and joblib on standard CPU hardware without a GPU. Table 2 lists the experimental controls. The latency measurements compare the evaluated configurations in this environment; they are not hardware-independent throughput estimates.

Table 2: Main experimental controls.

Item	Configuration
Data split	Stratified 70% training / 15% validation / 15% test
Repetitions	Five random seeds: 42, 0, 1, 2, 3
Meta-feature cross-validation	Stratified five-fold
HHO	30 hawks; 100 maximum iterations
HHO retention controls	0.9 accuracy / 0.1 parsimony fitness weighting; 99.5% accuracy guard; 67% minimum retained-feature ratio
Base classifiers	RF, XGBoost, GB; 100 estimators each
GWO	20 wolves; 50 maximum iterations; 10^{-4} convergence threshold
Decision threshold	0.5
Outcomes	Accuracy, precision, recall, F1-score, latency, selected-feature count

4. Results and Discussion

System setup used Python with pandas, NumPy, scikit-learn, XGBoost, LightGBM, matplotlib, seaborn, and joblib on standard CPU hardware without GPU acceleration. Each configuration used the same stratified 70% training, 15% validation, and 15% test partitions, five-fold cross-validation for meta-feature generation, and random seeds 42, 0, 1, 2, and 3. HHO used 30 hawks

for up to 100 iterations, while GWO used 20 wolves for up to 50 iterations. Per-instance latency covers prediction only and excludes the offline HHO and GWO searches.

The evaluation reports classification accuracy, F1-score, selected-feature count, and inference latency for individual classifiers, conventional stacking, the replicated nine-classifier stack, the HHO–GWO stack, and component-removal ablations. Table 3 lists the five-seed mean results for the primary configurations.

4.1 Classification performance

The individual classifiers and conventional stacking provided strong reference performance on both datasets. On UCI, XGBoost was the strongest individual model by mean accuracy (94.49%), while the replicated nine-classifier stack produced the highest mean accuracy (94.87%) and F1-score (95.02%). The HHO–GWO stack achieved 94.40% accuracy and 94.53% F1-score. On Mendeley, Random Forest achieved 96.39% accuracy, unoptimised stacking achieved 96.40%, and the replicated nine-classifier stack again produced the highest mean score at 96.77% accuracy and 96.71% F1-score. The HHO–GWO stack achieved 96.38% accuracy and 96.31% F1-score.

Table 3: Mean performance across five random seeds.

Dataset	Configuration	Accuracy	F1-score	Latency (ms)	Selected features
UCI	Random Forest	94.31%	94.45%	0.401	30.0
UCI	XGBoost	94.49%	94.61%	0.053	30.0
UCI	Gradient Boosting	93.85%	93.99%	0.029	30.0
UCI	Unoptimised stacking	94.53%	94.68%	0.521	30.0
UCI	Replicated nine-classifier stack	94.87%	95.02%	1.924	30.0
UCI	HHO–GWO stack	94.40%	94.53%	0.465	23.2
Mendeley	Random Forest	96.39%	96.33%	0.289	87.0
Mendeley	XGBoost	96.23%	96.17%	0.071	87.0
Mendeley	Gradient Boosting	95.68%	95.60%	0.037	87.0
Mendeley	Unoptimised stacking	96.40%	96.33%	0.375	87.0
Mendeley	Replicated nine-classifier stack	96.77%	96.71%	1.457	87.0
Mendeley	HHO–GWO stack	96.38%	96.31%	0.271	65.0

The HHO–GWO stack was not the best configuration when raw accuracy alone was the selection criterion. The replicated nine-classifier stack exceeded it by 0.47 percentage points on UCI and 0.39 percentage points on Mendeley. Nevertheless, the HHO–GWO stack remained close to the best stacked result on both datasets while using a reduced three-classifier architecture. Figure 2 compares these performance differences across the evaluated configurations.

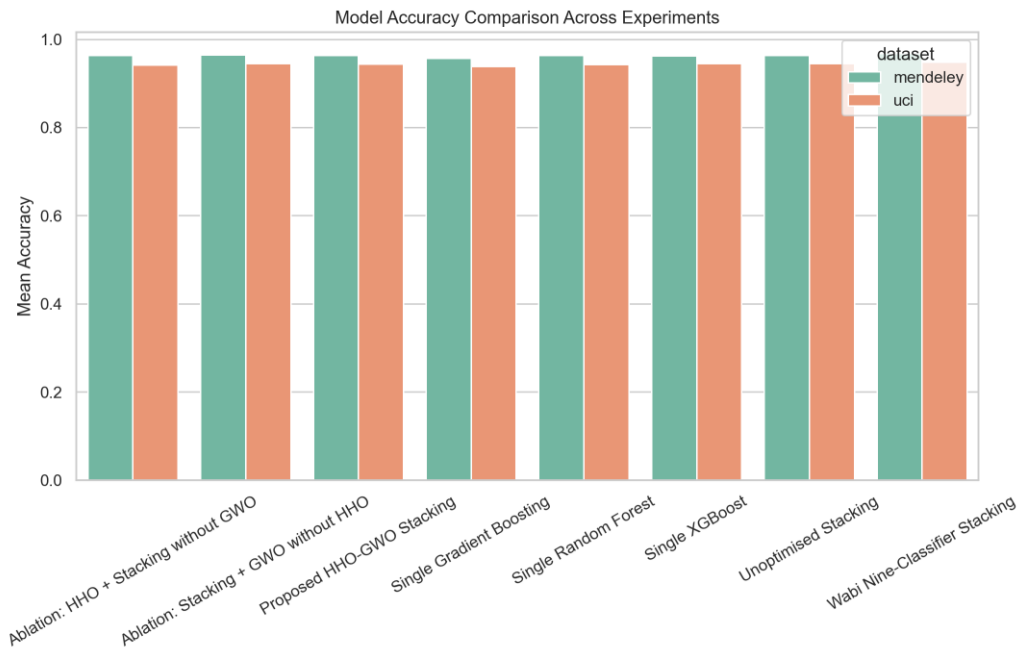


Figure 2: Mean classification performance across model configurations.

Table 3 and Figure 2 show that the HHO–GWO stack retains competitive accuracy and F1-score while reducing the number of model evaluations required for a final decision.

4.2 Feature efficiency

HHO reduced the UCI feature space from 30 to a mean of 23.2 selected features and the Mendeley space from 87 to a mean of 65.0 features. For the seed-42 run, the selected counts were 22 and 74, respectively. The retention guard and minimum ratio limit the reduction, preserving complementary URL, domain, and content indicators that may contribute to phishing classification. Figure 3 compares the original and HHO-selected feature counts.

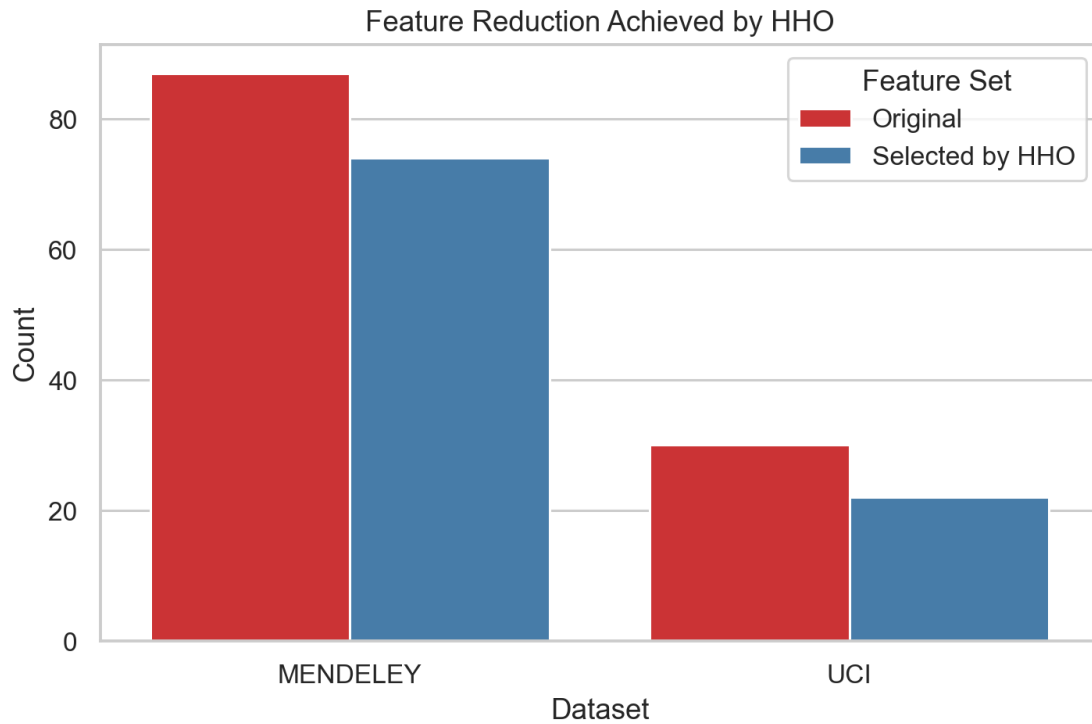


Figure 3: Feature reduction produced by HHO.

The ablation results quantify the effect of HHO. Removing HHO while retaining GWO increased the UCI feature count from 23.2 to 30.0 and the Mendeley count from 65.0 to 87.0. The full-feature configuration yielded marginally higher mean accuracy on both datasets—94.46% versus 94.40% on UCI and 96.42% versus 96.38% on Mendeley—but also higher latency: 0.580 ms versus 0.465 ms and 0.320 ms versus 0.271 ms, respectively. HHO produced the smaller representation at accuracy differences of 0.06 and 0.04 percentage points relative to the full-feature GWO configuration.

These patterns indicate that HHO contributed principally to a lower-dimensional inference representation. The resulting feature subsets remained predictive enough to keep the observed accuracy and F1-score close to the full-feature configurations.

4.3 Inference latency

The latency results distinguish the HHO–GWO stack most clearly from the complex stacking baseline. On UCI, it required 0.465 ms per instance, compared with 1.924 ms for the replicated nine-classifier stack. This is a reduction of 75.8%. On Mendeley, latency declined from 1.457 ms to 0.271 ms, a reduction of 81.4%. The reduced feature representations and the use of three rather than nine base classifiers both contribute to this difference. Figure 4 illustrates the latency difference between the two stacking designs.

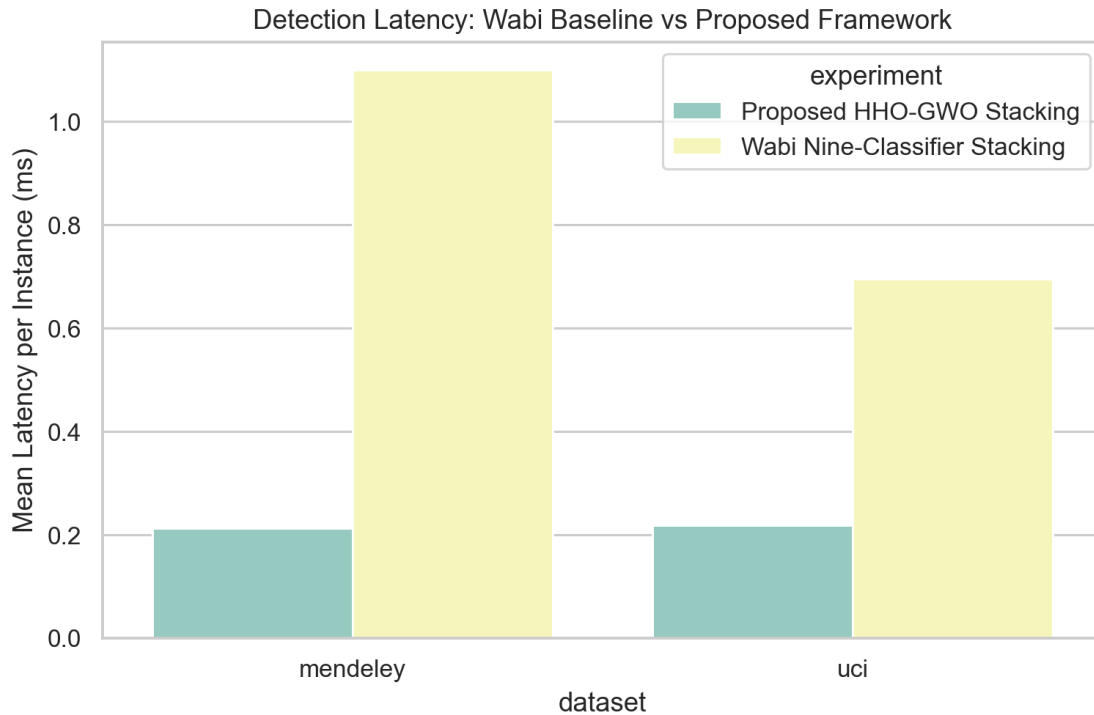


Figure 4: Mean inference-latency comparison between the replicated nine-classifier stack and the HHO–GWO stack.

The individual classifiers were faster than either stacking configuration because they required only one model evaluation. However, they do not provide the same probability-combination mechanism as the HHO–GWO stack. Among the stacking models, the HHO–GWO stack was faster than unoptimised stacking on UCI (0.465 ms versus 0.521 ms) and Mendeley (0.271 ms versus 0.375 ms). These results indicate that the reduced input representation and compact probability combination provide an efficiency advantage within the compared stacking designs.

4.4 Ablation and comparative analysis

The ablation analysis separates the roles of HHO and GWO. GWO optimised the contributions of the three base models after feature selection. In the seed-42 UCI run, the final weights were equal (0.3333 for RF, XGBoost, and GB). In the corresponding Mendeley run, RF received 0.8130, XGBoost received 0.1870, and GB received 0.0000. The validation search assigned different probability weights under the two data representations and splits. Table 4 presents the aggregate ablation results.

Table 4: Ablation results across five random seeds.

Dataset	Configuration	Accuracy	F1-score	Latency (ms)	Selected features
UCI	HHO–GWO stack	94.40%	94.53%	0.465	23.2
UCI	Stacking + GWO, without HHO	94.46%	94.59%	0.580	30.0
UCI	HHO + stacking, without GWO	94.24%	94.40%	0.504	23.2
Mendeley	HHO–GWO stack	96.38%	96.31%	0.271	65.0

Mendeley	Stacking + GWO, without HHO	96.42%	96.36%	0.320	87.0
Mendeley	HHO + stacking, without GWO	96.39%	96.33%	0.234	65.0

Figure 5 visualises the same component-removal comparison.

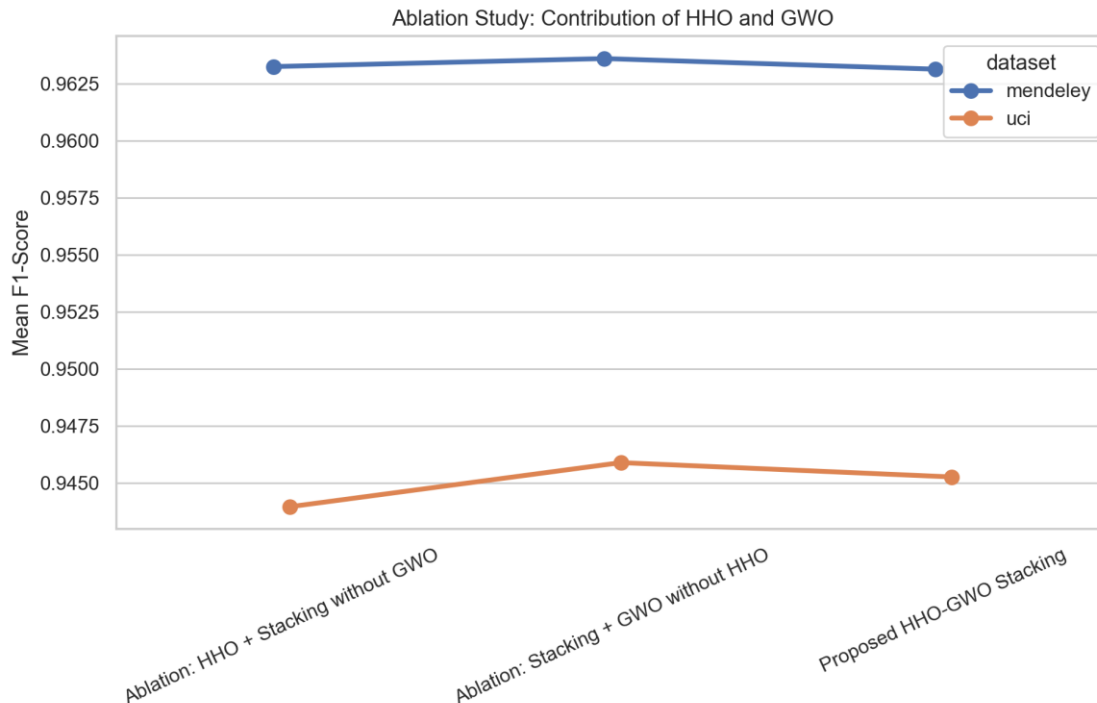


Figure 5: Ablation comparison of HHO and GWO components.

Removing GWO while retaining HHO reduced mean UCI accuracy from 94.40% to 94.24% and F1-score from 94.53% to 94.40%; latency increased slightly from 0.465 ms to 0.504 ms. On Mendeley, the no-GWO configuration was marginally higher in raw accuracy (96.39% versus 96.38%) and F1-score (96.33% versus 96.31%) and slightly lower in latency (0.234 ms versus 0.271 ms). GWO did not increase every metric on both datasets. It selects classifier contributions from validation data within the reduced stack and produced competitive aggregate performance across the two datasets.

The HHO–GWO stack retained the strongest overall balance of the three ablation configurations on UCI: it achieved the highest accuracy and F1-score while using the reduced feature set. On Mendeley, the no-HHO and no-GWO variants produced marginally higher accuracy values, but the differences were 0.04 and 0.01 percentage points, respectively. Table 4 and Figure 5 show that neither HHO nor GWO dominates every metric on both datasets. Together, they form a compact stacking configuration whose classification performance remains close to that of the larger baseline.

Compared with the replicated nine-classifier stack, the HHO–GWO stack uses one third of the base classifiers, 6.8 fewer features on average on UCI, and 22.0 fewer features on average on Mendeley. The resulting latency reductions are 75.8% and 81.4%, whereas the accuracy differences remain below half a percentage point. The comparison covers a common offline evaluation environment with pre-extracted benchmark features, not end-to-end deployment performance.

5. Conclusion

This paper evaluated an HHO–GWO phishing website classification model that integrates HHO feature selection, Random Forest, XGBoost, Gradient Boosting, and GWO-optimised probability weighting. Five-fold evaluation on the UCI Phishing Websites and Mendeley Web Page Phishing Detection datasets showed mean accuracies of 94.40% and 96.38% and mean F1-scores of 94.53% and 96.31%, respectively. The model reduced the average feature counts to 23.2 of 30 on UCI and 65.0 of 87 on Mendeley. Compared with the replicated nine-classifier stacking baseline, the HHO–GWO stack reduced mean inference latency from 1.924 ms to 0.465 ms on UCI and from 1.457 ms to 0.271 ms on Mendeley, while the corresponding accuracy differences were 0.47 and 0.39 percentage points.

The results show that a feature-reduced three-classifier stack can retain competitive benchmark performance while substantially lowering the measured classification cost of a larger stacking design. The present evaluation is limited to static datasets with pre-extracted features and offline inference; it does not establish end-to-end live-URL performance or robustness to temporally changing phishing campaigns. Future work should evaluate recent temporally separated datasets, include feature extraction in the latency measurement, and compare repeated runs using statistical tests.

References

- Hassan, A., Islam, M. M., & Islam, M. R. (2024). Phishing website identification: Unleashing the machine learning techniques. *Journal of Science and Technology*, 22.
- Bayraktar, C. (2026). Stacking-based model for phishing website detection system. *Afyon Kocatepe University Journal of Sciences and Engineering*, 26(1), 78–88.
- Pathak, P., & Shrivastava, A. K. (2024). Development of proposed model using random forest with optimization technique for classification of phishing website. *SN Computer Science*, 5(8), 1059.
- Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5(2), 241–259.
- Wabi, H. A., Ojeniyi, J. A., Idris, I., Subairu, S. O., & Al-Mekhlafi, Z. G. (2024). Stack ensemble model for detection of phishing website. In *2024 International Conference on Science, Engineering and Business for Driving Sustainable Development Goals (SEB4SDG)*, 1–6.
- Elsheh, M. M., & Abolawaifa, E. (2025). Hybrid stacking ensemble model for phishing URL detection using PCA and machine learning. *Journal of Technology Research*, 3(Special Issue), 515–525.
- Bari, N., Saleem, T., Shah, M., Algarni, A., Patel, A., & Ullah, I. (2025). A filter-based feature selection framework to detect phishing URLs using stacking ensemble machine learning. *Computer Modeling in Engineering & Sciences*, 145(1), 1167–1187.
- Kanathey, K., Gupta, V., & Imam, F. (2025). An enhanced and optimized stacking ensemble framework for phishing URLs detection. In *2025 4th OPJU International Technology Conference (OTCON)*, 1–6.

- Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems*, 97, 849–872.
- Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, 69, 46–61.
- Dua, D., & Graff, C. (2019). UCI Machine Learning Repository. *University of California, Irvine, School of Information and Computer Sciences*.
- Hannousse, A., & Yahiouche, S. (2021). Towards benchmark datasets for machine learning based website phishing detection: An experimental study. *Engineering Applications of Artificial Intelligence*, 104, 104347.
- Abualhaj, M. M., Abu-Shareha, A. A., Alkhatib, S. N., Shambour, Q. Y., & Alsaaidah, A. M. (2025). Detecting spam using Harris Hawks optimizer as a feature selection algorithm. *Bulletin of Electrical Engineering and Informatics*, 14(3), 2361–2369.
- Hamdan, A., Tahboush, M., Adawy, M., Alwada'n, T., Ghwanmeh, S., & Husni, M. (2024). Phishing detection using grey wolf and particle swarm optimizer. *International Journal of Electrical and Computer Engineering*, 14(5), 5961–5969.
- Ravula, V., & Ramaiah, M. (2025). Enhancing phishing detection with dynamic optimization and character-level deep learning in cloud environments. *PeerJ Computer Science*, 11, e2640.