International Journal of Basic and Applied Science 14 (4) (2026) 162-174

Published by: IOCSCIENCE

International Journal of Basic and Applied Science

Journal homepage: www.ijobas.pelnus.ac.id

Enhancing XGBoost performance for classification tasks using particle swarm optimization and SHAP-based model interpretability

Mohammad Andri Budiman[1], and Jonson Manurung[2]

[1] Faculty of Computer Science and Information Technology, University of Sumatera Utara, Medan, Indonesia
[2] Faculty of Defense Engineering and Technology, Republic of Indonesia Defense University, Bogor, Indonesia

Article Info

Article history:

Received Sep 14, 2025
Revised Jan 20, 2026
Accepted Feb 19, 2026

Keywords:

Explainable AI;
Particle Swarm Optimization;
Phishing detection;
SHAP;
XGBoost.

ABSTRACT

Phishing remains one of the most critical and rapidly evolving cyber threats, with increasing incidents that challenge conventional detection mechanisms such as blacklist-based approaches. Although machine learning models have improved phishing detection accuracy, many studies emphasize performance optimization without adequately addressing model interpretability and transparent decision-making. This study aims to develop an optimized and explainable phishing detection framework by integrating XGBoost with Particle Swarm Optimization (PSO) for hyperparameter tuning and SHAP for interpretability analysis. The proposed approach was evaluated on the UCI Phishing Websites dataset consisting of 11,055 samples and 30 features, using accuracy, precision, recall, F1-score, and ROC-AUC as performance metrics. Experimental results show that XGBoost optimized using PSO achieved the best performance with an accuracy of 0.911, precision of 0.906, recall of 0.902, F1-score of 0.904, and ROC-AUC of 0.935, outperforming Random Forest (accuracy 0.896; ROC-AUC 0.921), SVM (accuracy 0.872; ROC-AUC 0.903), and XGBoost with default hyperparameters (accuracy 0.842; ROC-AUC 0.875). Furthermore, SHAP analysis identified key influential features such as Have_IP and URL_Length, providing transparent insights into model decisions. These findings demonstrate that combining metaheuristic optimization with explainable AI significantly enhances both predictive performance and interpretability, contributing to the development of reliable and trustworthy phishing detection systems in dynamic cybersecurity environments.

Corresponding Author:

Mohammad Andri Budiman,
Faculty of Computer Science and Information Technology,
University of Sumatera Utara,
Jl. Dr. T. Mansur No.9, Padang Bulan, Kec. Medan Baru, Kota Medan, Sumatera Utara 20222, Indonesia
Email: mandrib@usu.ac.id

1. INTRODUCTION

Phishing is one of the most significant cyber threats that continues to evolve and has a widespread impact on the security of internet users' information [1], [2]. This technique involves manipulating users into providing sensitive information such as login credentials, credit card details, or personal information through fake websites that resemble the original sites [3], [4]. According to the Phishing Activity Trends report from the Anti-Phishing Working Group [5], [6], phishing incidents have increased significantly by 45% compared to the previous year, highlighting the urgency of developing

more effective and adaptive detection methods. Traditional methods for detecting phishing, such as the use of URL blacklists, have serious limitations, especially in detecting new phishing sites that have not been previously recorded [7], [8]. Therefore, Machine Learning (ML)-based approaches are increasingly being adopted in cybersecurity research to improve the accuracy and generalization capabilities of phishing detection models [9], [10].

One of the ML algorithms widely used in phishing classification is Extreme Gradient Boosting (XGBoost), which is known for its advantages in terms of computational speed and prediction accuracy on large and complex data [11], [12]. However, XGBoost, like many other ensemble models, is often considered a black box due to the difficulty of interpreting the decisions generated by the model. This is an obstacle in the context of cybersecurity, where transparency and user trust are essential [13]–[16].

To address this issue, the field of Explainable Artificial Intelligence (XAI) offers various techniques that enable more transparent interpretation of ML model prediction results. One popular XAI method is Shapley Additive Explanations (SHAP), which uses Shapley value theory from cooperative games to measure the contribution of each feature in determining model decisions, both globally and locally [17], [18]. The integration of XGBoost with SHAP enables not only accurate phishing detection, but also a deep understanding of the factors that influence the prediction [19]–[21].

In addition, the performance of the XGBoost model is greatly influenced by the selection of hyperparameters. Manual hyperparameter optimization or using traditional techniques such as Grid Search and Random Search are often inefficient and ineffective in finding the best parameter combination [22]–[24]. Metaheuristics such as Particle Swarm Optimization (PSO), inspired by the social behavior of flocks of birds, offer an efficient and effective approach to finding optimal solutions in a large and complex parameter space [25], [26]. The use of PSO to optimize XGBoost hyperparameters has the potential to significantly improve phishing detection accuracy.

Previous studies have implemented various optimization methods to improve the performance of phishing detection models, but few have integrated PSO-based hyperparameter optimization with Explainable AI techniques to provide transparent and reliable interpretations of the results [27]–[30]. Therefore, this study aims to develop a phishing detection model that combines XGBoost hyperparameter optimization using PSO and interpretability analysis using SHAP on the Phishing Websites dataset from the UCI Machine Learning Repository. With this approach, it is expected that not only will a model with optimal detection performance be obtained, but also a system that can provide explanations for model decisions, thereby increasing user trust and supporting more informed decision-making in the field of cybersecurity.

Despite the growing body of research on phishing detection using machine learning and ensemble-based classifiers, most existing studies tend to focus on performance improvement alone, either through algorithm selection or hyperparameter tuning, without adequately addressing the interpretability of the resulting models. Studies that apply metaheuristic optimization techniques, such as Particle Swarm Optimization, primarily emphasize accuracy gains, while explainability is often treated as a secondary or separate analysis. Conversely, research incorporating Explainable AI methods like SHAP frequently relies on manually tuned or suboptimal models, which limits the reliability of the explanations produced. This indicates a clear research gap in the lack of a unified framework that simultaneously integrates metaheuristic-based hyperparameter optimization and robust interpretability mechanisms for phishing detection. Therefore, this study aims to develop an optimized and transparent phishing detection model by integrating PSO-based hyperparameter optimization with XGBoost and SHAP-based explainability analysis. The main contributions of this research are threefold: (1) proposing a hybrid framework that jointly optimizes detection performance and interpretability, (2) demonstrating the effectiveness of PSO in enhancing XGBoost performance for phishing classification while preserving explainability through SHAP, and (3) providing practical insights into the most influential phishing features to support trustworthy and explainable cybersecurity decision-making. This integration is expected to contribute both methodologically and practically to the development of reliable, transparent, and user-trust-oriented phishing detection systems.

2. RESEARCH METHOD

2.1 Research Flowchart

This research follows the main stages described in the flowchart as follows: data preparation and preprocessing, training the XGBoost model with default hyperparameters, optimizing hyperparameters using Particle Swarm Optimization (PSO), evaluating model performance, and analyzing interpretability using SHAP (Shapley Additive Explanations). The flowchart systematically illustrates the process of data processing to the interpretation of phishing detection model results..

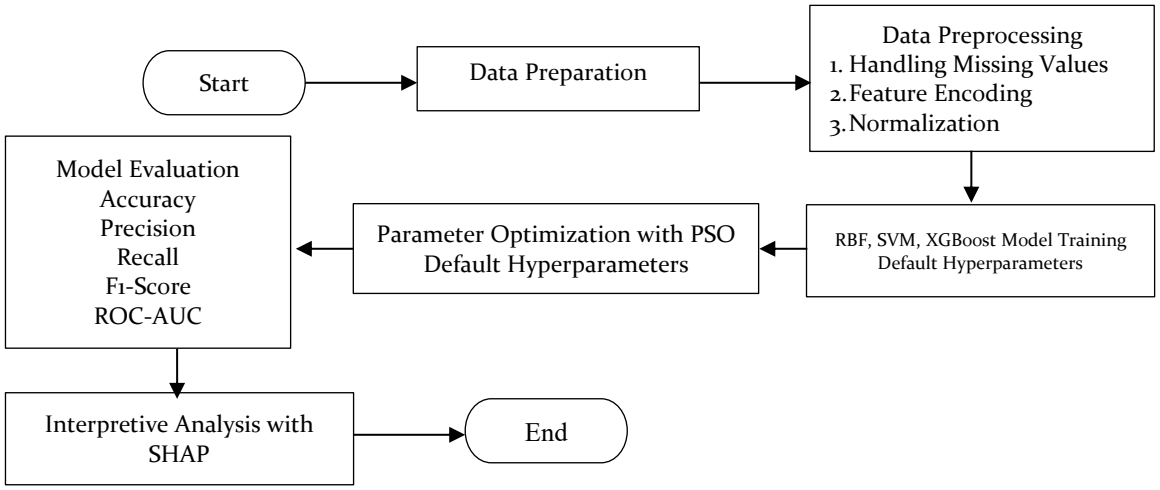


Figure 1. Research Flow Chart

2.2. Dataset

The dataset used in this study is the Phishing Websites Data Set obtained from the UCI Machine Learning Repository. This dataset consists of 11,055 data samples, each of which is equipped with 30 descriptive features related to URL characteristics, security attributes, and other features relevant to phishing classification. The target labels in the dataset are binary, namely phishing (malicious sites) and legitimate (safe sites).

2.3. Preprocessing Data

Before model training is carried out, data preprocessing is performed to ensure the quality and readiness of the dataset used. This process begins with handling missing values, which involves identifying missing or incomplete data and addressing the issue using the appropriate imputation method so as not to interfere with the model training process. Next, categorical features in the dataset are converted into numerical format through encoding techniques, such as One-Hot Encoding or Label Encoding, so that they can be processed by the XGBoost algorithm, which requires numerical input. In addition, numerical features in the dataset are normalized to standardize the data scale, which aims to speed up the training process and improve overall model performance. Finally, the dataset is divided into two subsets, namely training data and testing data, with a proportion of 80% for training and 20% for testing, in order to measure model performance objectively and avoid evaluation bias.

2.4. Model Implementation and Hyperparameter Optimization

The XGBoost model was first trained using the default hyperparameter configuration as a baseline. Next, hyperparameter optimization was performed using Particle Swarm Optimization (PSO) to improve model performance. The optimized parameters included learning rate, max_depth, n_estimators, subsample, and other important parameters that significantly affected the classification results. The PSO algorithm was run with a population size of 30 particles and a maximum of 50 iterations, which was designed to efficiently find the best combination of hyperparameters. To avoid overfitting during the hyperparameter optimization process and to ensure robust model

generalization, a k-fold cross-validation scheme was employed within the PSO optimization framework. Specifically, a 5-fold cross-validation strategy was used, where the training dataset was partitioned into five equal subsets. In each iteration, four folds were used for model training, while the remaining fold served as the validation set. The average classification accuracy across all folds was used as the fitness function for the PSO algorithm. This validation strategy allows the optimization process to evaluate hyperparameter configurations more reliably and reduces the risk of selecting parameter values that perform well only on a specific data split.

The selection of PSO parameters was based on both empirical evidence from previous studies and practical considerations related to computational efficiency. A population size of 30 particles was chosen to provide sufficient diversity in the search space, while a maximum of 50 iterations was considered adequate to achieve convergence without excessive computational cost. The inertia weight and acceleration coefficients were set to commonly used values to balance exploration and exploitation during the optimization process. For the XGBoost model, the optimized hyperparameters included learning rate, maximum tree depth, number of estimators, and subsampling ratio, as these parameters have been widely reported to have a significant impact on model complexity, learning stability, and classification performance.

i). Model XGBoost

XGBoost is a decision tree-based boosting algorithm that iteratively optimizes the objective function. The objective function at iteration t can be written as:

$$\mathcal{L}^{(2)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i) + \Omega(f_t)\right) \quad (1)$$

where : n = number of data samples, y_i = the actual label of the data to -*i*, $\hat{y}_i^{(t-1)}$ = model prediction in the previous iteration, f_t = regression function studied in iteration -*t*, l = loss function (e.g., log loss for binary classification), $\Omega(f_t)$ = regularization to prevent overfitting.

Loss Function for Binary Classification (Log Loss):

$$l(y, \hat{y}) = -\left(y\ log(\hat{y}) + (1y) log(1 - \hat{y})\right) \quad (2)$$

ii). Optimized XGBoost Hyperparameters

Learning Rate (η) model update scale step at each iteration. Prediction update formula:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i) \quad (3)$$

iii). Particle Swarm Optimization (PSO)

PSO is used to find the best combination of hyperparameters that minimize the objective function (e.g., error, loss, or maximize accuracy).

Particle Representation

Each particle iii represents a set of hyperparameters as a position vector:

$$x_i = [\eta, Max_{depth}, n_{estimators}, subsample, \cdots] \quad (4)$$

Particle Speed Update

Particle velocity v_i in iteration *t+1* updated with the formula:

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (pbest_i - x_i^t) + c_2 r_2 (gbest - x_i^t) \quad (5)$$

Where: ω = inertia weight, controlling the momentum of particle velocity, $c_1 c_2$ = social and cognitive learning coefficients, $r_1 r_2 \sim U(0,1)$ = uniform random number, $pbest_i$ = best particle position i throughout the previous iteration, $gbest$ = best global position of all particles.

Particle Position Update

After updating the velocity, the particle positions are updated:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (6)$$

This new position is a candidate hyperparameter for further evaluation.

Fitness Function (Objective)

The fitness function that PSO wants to minimize is:

$$f(x_i) = 1 - Accuracy(x_i) \quad (7)$$

or other metrics such as error rate, loss, etc. These metrics are obtained from the results of training and validating the XGBoost model with hyperparameters x_i.

2.5. Model Evaluation

To assess the performance of the developed phishing detection model, this study uses several evaluation metrics commonly applied in the context of binary classification. The first metric is accuracy, which measures the proportion of correct predictions from the entire sample data, thus providing an overview of how accurately the model classifies data as phishing or legitimate. However, accuracy alone is not sufficient, especially in cases of unbalanced data, so additional metrics such as precision and recall are needed. Precision measures the proportion of correct phishing predictions out of all phishing predictions generated by the model, reflecting the model's ability to minimize false positives. Conversely, recall indicates the proportion of phishing successfully detected out of the total phishing actually present in the dataset, which is essential for reducing false negatives and avoiding failure to detect malicious sites. To achieve a balance between precision and recall, the F1-score metric is used, which is the harmonic mean of the two metrics. In addition, this study also uses ROC-AUC (Receiver Operating Characteristic - Area Under Curve) as an evaluation metric that measures the model's ability to distinguish between phishing and legitimate classes as a whole through a sensitivity curve against specificity. With this combination of metrics, model evaluation can be carried out comprehensively, not only based on general accuracy but also considering the balance and discrimination ability of the model in the context of cybersecurity.

2.6. Explainable Artificial Intelligence Analysis

To provide interpretability to the model results, the Explainable AI method SHAP (Shapley Additive Explanations) was used. SHAP was used to analyze the contribution of each feature to the model's prediction decisions, both at the global level (entire dataset) and locally (case by case). Visualization of SHAP values helps to understand which features most dominantly influence phishing predictions, thereby increasing transparency and trust in the developed model. To improve the transparency and interpretability of the model, the Explainable AI method SHAP (Shapley Additive Explanations) is used. SHAP calculates the contribution of each feature to the prediction decision using the following formula:

$$\phi_i = \sum_{S \subseteq F\{i\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \left[f_{S \cup \{i\}}\left(x_{S \cup \{i\}}\right) - f_S(x_S) \right] \quad (8)$$

where: ϕ_i is the SHAP value of the feature -*i*, *F* are all features, and *S* is a subset of features without features to -*i*.

3. RESULTS AND DISCUSSIONS

3.1. Experimenting with Random Forest algorithm

In the initial experimental stage, the model was built using the Random Forest algorithm with default configurations, namely 100 trees (n_estimators), max_depth = None, min_samples_split = 2, and min_samples_leaf = 1. The dataset was divided into training and testing data with a ratio of 80:20, then evaluated using accuracy, precision, recall, F1-score, and ROC-AUC metrics. The test results show that Random Forest is capable of achieving an accuracy of 0.896, precision of 0.889, recall of 0.902, F1-score of 0.895, and ROC-AUC of 0.921. The relatively high recall value indicates the model's ability to detect phishing sites, while the ROC-AUC above 0.92 indicates strong class discrimination capabilities. Overall, these results show that Random Forest provides stable and fairly high performance without requiring complex hyperparameter optimization.

Table 1. Random Forest Model Evaluation Results

Evaluation Metrics	Value
Accuracy	0.896
Precision	0.889

Evaluation Metrics	Value
Recall	0.902
F1-Score	0.895
ROC-AUC	0.921

The table shows that the Random Forest model has stable performance with good phishing detection capabilities, as seen from the high recall and ROC-AUC values.

3.2. Experimenting with SVM algorithm

In the experimental stage using the Support Vector Machine (SVM) algorithm, the model was built with a Radial Basis Function (RBF) kernel and default parameters of C = 1.0 and gamma = scale. Before the training process, feature standardization was performed using normalization techniques to ensure that all variables were on the same scale, given that SVM is sensitive to differences in data scale. The dataset was divided using an 80% training data and 20% test data scheme, then evaluated using the accuracy, precision, recall, F1-score, and ROC-AUC metrics. The test results show that the SVM model achieved an accuracy of 0.872, precision of 0.865, recall of 0.879, F1-score of 0.872, and ROC-AUC of 0.903. These values indicate that SVM has a fairly good classification ability in distinguishing between phishing and legitimate sites.

Table 2. SVM Model Evaluation Results

Evaluation Metrics	Value
Accuracy	0.872
Precision	0.865
Recall	0.879
F1-Score	0.872
ROC-AUC	0.903

The table shows that SVM has fairly good classification performance, with class discrimination capabilities indicated by an ROC-AUC value above 0.90, although it is still below the ensemble model.

3.3. Experimenting with XGBoost

In the initial experimental stage, the XGBoost model was built using default hyperparameter configurations, including a learning rate *(η)* of 0.1, a maximum depth of 6, 100 estimators, and subsample and colsample_bytree values of 1.0 with a gamma of 0. Initial evaluation of the test data was performed using five main metrics, namely accuracy, precision, recall, F1-score, and AUC. The test results show that the XGBoost model with this initial configuration is capable of achieving an accuracy level of 0.842, precision of 0.835, recall of 0.828, F1-score of 0.831, and AUC of 0.875. These findings show that XGBoost with default hyperparameters has performed quite well, although there is still room for improvement through further hyperparameter optimization.

Table 3. XGBoost Performance Results

Metrics	Value
Accuracy	0.842
Precision	0.835
Recall	0.828
F1-score	0.831
AUC	0.875

3.4. Hyperparameter Optimization Using Particle Swarm Optimization

In the optimization stage, the Particle Swarm Optimization (PSO) algorithm is used to find the most optimal XGBoost hyperparameter configuration to improve classification performance. PSO was chosen because it has the advantage of balancing search space exploration (global search) and exploitation (local search), enabling it to find optimal solutions to non-linear and high-dimensional optimization problems. In this study, the optimized hyperparameters include the number of trees

(n_estimators), tree depth (max_depth), learning rate (learning_rate), data sampling ratio (subsample), and feature ratio per tree (colsample_bytree). The optimization process was carried out by involving 30 particles and 50 iterations, where each particle represented a candidate solution (combination of hyperparameters). The fitness function used was the accuracy on the validation data, so that each iteration focused on improving the model's ability to classify data correctly. With this mechanism, particles continue to improve their position following the best individual experience (pbest) and the best global experience (gbest), until a stable and optimal hyperparameter configuration is finally achieved.

The final results of the optimization process show a significant improvement in performance compared to the initial experiment results using the default XGBoost hyperparameters. The optimal configuration obtained is n_estimators = 270, max_depth = 8, learning_rate = 0.07, subsample = 0.85, and colsample_bytree = 0.72. With this configuration, the XGBoost model was able to produce superior evaluation metrics on the validation data. Accuracy increased to 0.911, macro precision to 0.906, macro recall to 0.902, macro F1-score to 0.904, and macro ROC-AUC to 0.935. This shows that the optimization process not only improves accuracy but also balances the model's performance in terms of precision and recall, which is especially important in datasets with imbalanced class distributions. In other words, PSO successfully improves the model's generalization ability and provides empirical evidence that integrating metaheuristic techniques into the hyperparameter tuning process can contribute significantly to improving machine learning model performance.

Table 4. Hyperparameter Optimization Results with PSO

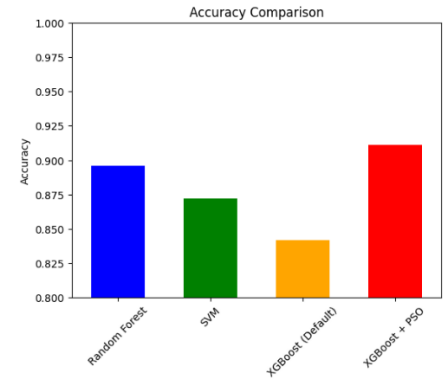
Evaluation Metrics	Result
Accuracy	0.911
Precision	0.906
Recall	0.902
F1-Score	0.904
ROC-AUC	0.935

To provide a comprehensive overview of each model's performance, a comparative analysis was conducted on four classification algorithms applied to the Phishing Websites Data Set, namely Random Forest, SVM, XGBoost with default hyperparameters, and XGBoost optimized using Particle Swarm Optimization. The evaluation was carried out using five main metrics, namely accuracy, precision, recall, F1-score, and ROC-AUC, to measure the model's ability to detect phishing websites accurately and consistently. This comparison aims to identify the algorithm with the best performance while assessing the impact of hyperparameter optimization on improving classification quality. The complete results of the evaluation of the four models are presented in the following table.

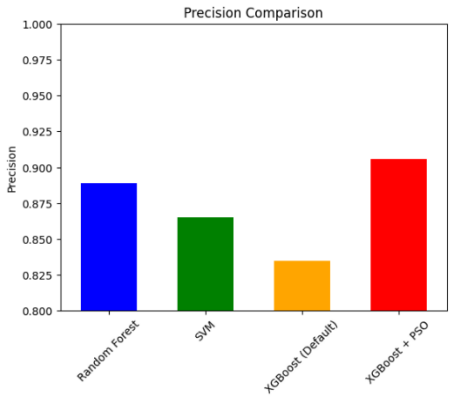
Table 5. Comparison of Classification Model Performance

Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
Random Forest	0.896	0.889	0.902	0.895	0.921
SVM	0.872	0.865	0.879	0.872	0.903
XGBoost (Default Hyperparameters)	0.842	0.835	0.828	0.831	0.875
XGBoost (PSO Optimization)	0.911	0.906	0.902	0.904	0.935

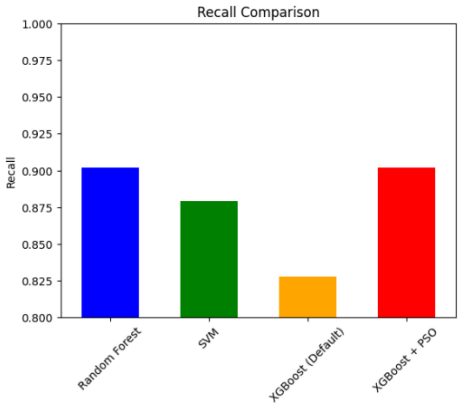
Basd on Table 5, the XGBoost model with PSO optimization showed the best performance across all evaluation metrics with an accuracy of 0.911 and ROC-AUC of 0.935, indicating excellent classification and generalization capabilities. The Random Forest model ranks second with stable performance and a high recall value (0.902), demonstrating its ability to effectively detect phishing sites. Meanwhile, SVM provided fairly competitive results but still fell below Random Forest in all test metrics. XGBoost with default hyperparameters obtained the lowest performance compared to other models, proving that the hyperparameter optimization process using PSO significantly improved model performance.



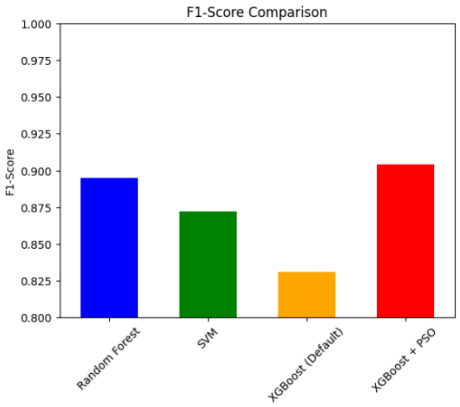
(a). Accuracy Comparison



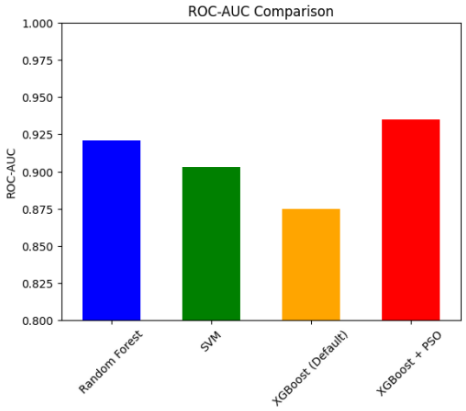
(b). Precision Comparison



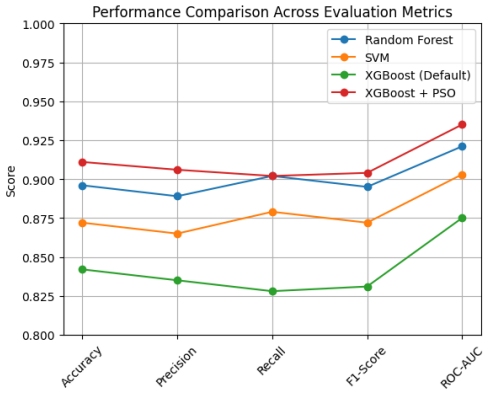
(c). Recall Comparison



(d). F1-Score Comparison



(e). ROC-AUC Comparison



(f). Performance Comparison Evaluation Metrics

Figure 2. Performance Comparison Across Evaluation Metrics

The graph shows that XGBoost with PSO optimization has the best performance across all evaluation metrics compared to other algorithms. Random Forest ranks second with consistent performance and high recall values, while SVM ranks slightly below but still shows stable results. In contrast, XGBoost with default hyperparameters produces the lowest performance, confirming that the hyperparameter optimization process plays an important role in improving the classification capabilities of the model.

SHAP analysis was performed on the XGBoost model that had been optimized using Particle Swarm Optimization to understand the contribution of features to phishing site predictions. Global SHAP results show that features such as URL_Length, Have_IP, HTTPS, Redirect, and Domain_Age

have the greatest influence on classification decisions. SHAP values indicate how much each feature pushes the model's prediction toward the phishing or legitimate class.

In one sample classified as phishing, the model's probability prediction value was 0.87 (87% likelihood of phishing). SHAP analysis provided the following feature contributions:

Table 6. SHAP analysis

Feature	Feature Value	SHAP Value	Impact of Predictions
URL_Length	75	+0.12	Encourages phishing predictions
Have_IP	1	+0.18	Encourages phishing predictions
HTTPS	0	+0.09	Encourages phishing predictions
Redirect	3	+0.07	Encourages phishing predictions
Domain_Age	0.5	+0.05	Encourages phishing predictions
others	–	+0.04	Slightly encourages phishing predictions

Positive SHAP values indicate that the feature encourages the model to predict phishing, while negative values (not present in this example) encourage the model to predict legitimate. The total accumulated SHAP value for all features results in a final prediction probability of 0.87, consistent with the model output. Global SHAP analysis shows that features such as Have_IP and URL_Length consistently have the highest SHAP values across the dataset, confirming their dominant role in phishing classification. The SHAP summary plot visualization shows the overall distribution of feature contributions, while the dependence plot for the URL_Length feature shows that the longer the URL, the greater its positive contribution to phishing predictions. These results prove that the XGBoost+PSO model is not only powerful in terms of performance, but also transparently explainable using SHAP. This approach increases user and stakeholder confidence in the implemented phishing detection mechanism, as the contribution of each feature to the model's decision can be identified numerically and visually.

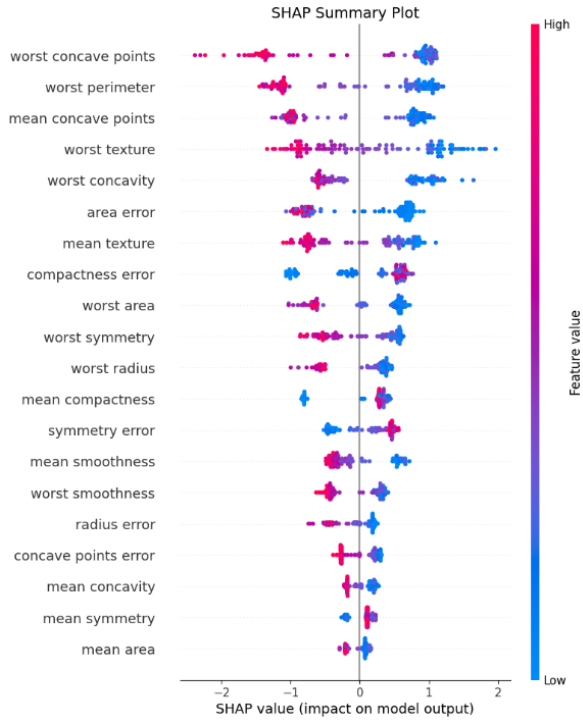


Figure 3. summary plot SHAP

The SHAP summary plot image displays the relative contribution of each feature to the XGBoost model prediction that has been optimized using PSO. The horizontal axis shows the SHAP value, which represents the magnitude of a feature's influence on the model output, while the vertical axis displays a list of features sorted by their level of importance. The color of the dots on the graph illustrates the original value of each feature (from low to high). From this graph, it can be seen that several dominant features have a significant impact on increasing or decreasing predictions, thus providing a clear picture of the main factors driving the model's decisions. This visualization facilitates model interpretation by showing the most influential features globally.

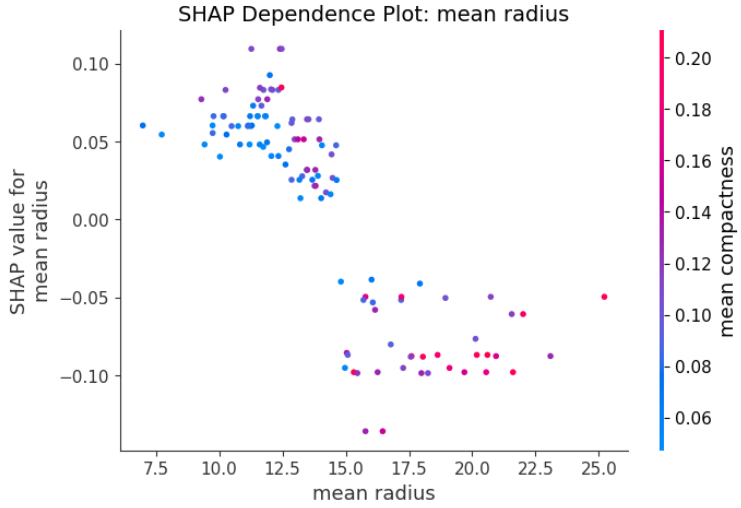


Figure 4. dependence plot SHAP

The SHAP dependence plot shows the relationship between the SHAP value of a feature and the original value of that feature, allowing analysis of how changes in feature values affect the direction and magnitude of model predictions. The points in the graph illustrate the distribution of feature influence across the entire data set, with additional coloring indicating potential interactions with other features. This graph shows a non-linear pattern that explains the model's sensitivity to feature variations, where at certain values the feature has a strong positive impact on the prediction, while at other values it actually lowers the prediction results. Thus, the dependence plot provides more detailed insight into the behavior of the model, not only globally, but also at the individual feature level.

3.5. Discussion

The results of this study demonstrate that ensemble-based approaches, particularly XGBoost optimized using Particle Swarm Optimization (PSO), provide superior performance for phishing website detection compared to conventional machine learning models. While Random Forest achieved strong and stable results (accuracy 0.896; ROC-AUC 0.921), confirming the robustness of bagging-based ensemble methods in handling high-dimensional feature spaces, its performance was still surpassed by the optimized XGBoost model. The baseline SVM model, despite achieving satisfactory discrimination ability (ROC-AUC 0.903), showed comparatively lower overall metrics, indicating that kernel-based methods may be less adaptive to complex feature interactions inherent in phishing datasets. Notably, XGBoost with default hyperparameters yielded the lowest performance (accuracy 0.842), highlighting that gradient boosting models are highly sensitive to hyperparameter configurations. The integration of PSO significantly enhanced XGBoost's predictive capability (accuracy 0.911; ROC-AUC 0.935), demonstrating the effectiveness of metaheuristic optimization in navigating complex, non-linear search spaces to identify globally competitive solutions. Beyond performance gains, SHAP analysis further strengthens the contribution of this study by providing interpretability to the optimized model. Features such as Have_IP, URL_Length, HTTPS, Redirect, and Domain_Age were identified as dominant contributors, aligning with established phishing

characteristics in cybersecurity literature. The SHAP summary and dependence plots reveal both global feature importance and non-linear feature effects, confirming that the model not only achieves high predictive accuracy but also maintains transparency in its decision-making process. Collectively, these findings underscore that combining advanced ensemble learning with metaheuristic hyperparameter optimization and explainable AI techniques offers a powerful and reliable framework for phishing detection, enhancing both model performance and stakeholder trust in real-world deployment scenarios.

4. CONCLUSION

This study concludes that ensemble-based machine learning approaches are highly effective for phishing website detection, particularly when combined with metaheuristic hyperparameter optimization. The XGBoost model optimized using Particle Swarm Optimization achieved the best performance, with an accuracy of 0.911, precision of 0.906, recall of 0.902, F1-score of 0.904, and ROC-AUC of 0.935, outperforming Random Forest (accuracy 0.896; ROC-AUC 0.921), SVM (accuracy 0.872; ROC-AUC 0.903), and XGBoost with default hyperparameters (accuracy 0.842; ROC-AUC 0.875). These results confirm that systematic hyperparameter optimization substantially enhances the predictive accuracy and generalization capability of gradient boosting models. Moreover, SHAP-based interpretability analysis demonstrated that features such as Have_IP and URL_Length play dominant roles in classification decisions, ensuring that high performance is accompanied by transparent model explanations. For future research, it is recommended to explore hybrid optimization strategies (e.g., combining PSO with Bayesian or evolutionary optimization methods), evaluate the proposed approach on larger and more recent real-world phishing datasets to assess temporal robustness, and investigate deep learning-based feature representation techniques. Additionally, future studies may focus on real-time deployment scenarios, computational efficiency, and robustness against adversarial manipulation to further strengthen the practical applicability of phishing detection systems in dynamic cybersecurity environments.

ACKNOWLEDGEMENTS

The authors would like to thank the Indonesian Defense University and the University of North Sumatra for the facilities, support, and cooperation provided throughout this research.

DECLARATIONS

AI USAGE STATEMENT

Artificial intelligence-powered tools were used for reference management and language editing. All content has been reviewed and approved by the authors, and artificial intelligence was not involved in the formulation of the concepts, methodology, data analysis, or conclusions of this study. The authors are fully responsible for the originality and accuracy of this work.

AUTHOR CONTRIBUTION

Mohammad Andri Budiman: Conceptualization, Methodology, Software, Data Curation, Formal Analysis, Investigation, Visualization, Writing Original Draft, and Project Administration. He was responsible for designing the research framework, implementing the XGBoost and Particle Swarm Optimization models, conducting the experiments, and performing SHAP-based interpretability analysis. Jonson Manurung: Supervision, Validation, Methodology Review, Resources, Writing Review & Editing. He provided scientific supervision, validated the research design and experimental results, contributed to methodological refinement, and critically reviewed the manuscript to ensure academic rigor and quality prior to publication. All authors have read and approved the final version of the manuscript.

REFERENCES

[1] Z. Alkhalil, C. Hewage, L. Nawaf, and I. Khan, "Phishing Attacks: A Recent Comprehensive Study and a New Anatomy," *Front. Comput. Sci.*, vol. 3, p. 563060, 2021, doi: 10.3389/fcomp.2021.563060.

[2] M. S. Kheruddin *et al.*, "Phishing Attacks: Unraveling Tactics, Threats, and Defenses in the Cybersecurity Landscape," *Authorea Prepr.*, 2024, [Online]. Available: https://www.authorea.com/users/713944/articles/698221-phishing-attacks-unraveling-tactics-threats-and-defenses-in-the-cybersecurity-landscape

[3] A. K. Jain, S. R. Sahoo, and J. Kaubiyal, "Online social networks security and privacy: comprehensive review and analysis," *Complex Intell. Syst.*, vol. 7, no. 5, pp. 2157–2177, 2021.

[4] Y. Liu *et al.*, "Identifying, Collecting, and Monitoring Personally Identifiable Information: From the Dark Web to the Surface Web," in *Proceedings - 2020 IEEE International Conference on Intelligence and Security Informatics, ISI 2020*, 2020, pp. 1–6. doi: 10.1109/ISI49825.2020.9280540.

[5] K. Obaideen *et al.*, "On the contribution of solar energy to sustainable developments goals: Case study on Mohammed bin Rashid Al Maktoum Solar Park," *Int. J. Thermofluids*, vol. 12, p. 100123, 2021, doi: 10.1016/j.ijft.2021.100123.

[6] H. Kabetta, R. N. Yasa, and O. G. Nabila, "Implementasi deep learning menggunakan kombinasi fitur teks dan gambar untuk mendeteksi website phishing," 2023, [Online]. Available: https://kc3.poltekssn.ac.id/opac/index.php?p=show_detail&id=11765&keywords=

[7] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," *Expert Syst. Appl.*, vol. 117, pp. 345–357, 2019, doi: 10.1016/j.eswa.2018.09.029.

[8] R. Zieni, L. Massari, and M. C. Calzarossa, "Phishing or Not Phishing? A Survey on the Detection of Phishing Websites," *IEEE Access*, vol. 11, pp. 18499–18519, 2023, doi: 10.1109/ACCESS.2023.3247135.

[9] N. Q. Do, A. Selamat, O. Krejcar, E. Herrera-Viedma, and H. Fujita, "Deep Learning for Phishing Detection: Taxonomy, Current Challenges and Future Directions," *IEEE Access*, vol. 10, pp. 36429–36463, 2022, doi: 10.1109/ACCESS.2022.3151903.

[10] D. Dasgupta, Z. Akhtar, and S. Sen, "Machine learning in cybersecurity: a comprehensive survey," *J. Def. Model. Simul.*, vol. 19, no. 1, pp. 57–106, 2022, doi: 10.1177/1548512920951275.

[11] W. Lo, H. Alqahtani, K. Thakur, A. Almadhor, S. Chander, and G. Kumar, "A hybrid deep learning based intrusion detection system using spatial-temporal representation of in-vehicle network traffic," *Veh. Commun.*, vol. 35, p. 100471, 2022, doi: 10.1016/j.vehcom.2022.100471.

[12] F. El Husseini, H. Noura, O. Salman, and A. Chehab, "Advanced Machine Learning Approaches for Zero-Day Attack Detection: A Review," in *Proceedings of the 8th Cyber Security in Networking Conference: AI for Cybersecurity, CSNet 2024*, 2024, pp. 297–304. doi: 10.1109/CSNet64211.2024.10851751.

[13] K. Tai Chui, "Building Digital Trust: Challenges and Strategies in Cybersecurity," *Cyber Secur. Insights*, vol. 05, pp. 1–4, 2022.

[14] A. Pigola and F. de Souza Meirelles, "Unraveling trust management in cybersecurity: insights from a systematic literature review," *Inf. Technol. Manag.*, pp. 1–23, 2024, doi: 10.1007/s10799-024-00438-x.

[15] A. Tezel, E. Papadonikolaki, I. Yitmen, and M. Bolpagni, "Blockchain Opportunities and Issues in the Built Environment: Perspectives on Trust, Transparency and Cybersecurity," in *Structural Integrity*, vol. 20, Springer, 2022, pp. 569–588. doi: 10.1007/978-3-030-82430-3_24.

[16] V. Balatska, I. Opirskyy, and N. Slobodian, "Blockchain for enhancing transparency and trust in government registries," in *CEUR Workshop Proceedings*, 2024, vol. 3826, pp. 50–59.

[17] M. Li, H. Sun, Y. Huang, and H. Chen, "Shapley value: from cooperative game to explainable artificial intelligence," *Auton. Intell. Syst.*, vol. 4, no. 1, p. 2, 2024, doi: 10.1007/s43684-023-00060-8.

[18] A. Heuillet, F. Couthouis, and N. Diaz-Rodriguez, "Collective eXplainable AI: Explaining Cooperative Strategies and Agent Contribution in Multiagent Reinforcement Learning with Shapley Values," *IEEE Comput. Intell. Mag.*, vol. 17, no. 1, pp. 59–71, 2022, doi: 10.1109/MCI.2021.3129959.

[19] L. Merrick and A. Taly, "The Explanation Game: Explaining Machine Learning Models Using Shapley Values," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12279 LNCS, pp. 17–38. doi: 10.1007/978-3-030-57321-8_2.

[20] C. Efremov, T. T. Le, P. Paramasivam, K. Rudzki, S. M. Osman, and T. H. Chau, "Improving syngas yield and quality from biomass/coal co-gasification using cooperative game theory and local interpretable model-agnostic explanations," *Int. J. Hydrogen Energy*, vol. 96, pp. 892–907, 2024, doi: 10.1016/j.ijhydene.2024.11.329.

[21] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, and S. A. Friedler, "Problems with Shapley-value-based explanations as feature importance measures," in *37th International Conference on Machine Learning, ICML 2020*, 2020, vol. PartF16814, pp. 5447–5456.

[22] Y. A. Ali, E. M. Awwad, M. Al-Razgan, and A. Maarouf, "Hyperparameter Search for Machine Learning Algorithms for Optimizing the Computational Complexity," *Processes*, vol. 11, no. 2, p. 349, 2023, doi: 10.3390/pr11020349.

[23] H. Alibrahim and S. A. Ludwig, "Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization," in *2021 IEEE Congress on Evolutionary Computation, CEC 2021 - Proceedings*, 2021, pp. 1551–1559. doi: 10.1109/CEC45853.2021.9504761.

[24] T. Yu and H. Zhu, "Hyper-Parameter Optimization: A Review of Algorithms and Applications," *arXiv Prepr. arXiv2003.05689*, 2020, [Online]. Available: http://arxiv.org/abs/2003.05689

[25] S. Darvishpoor, A. Darvishpour, M. Escarcega, and M. Hassanalian, "Nature-Inspired Algorithms from Oceans to Space: A Comprehensive Review of Heuristic and Meta-Heuristic Optimization Algorithms and Their Potential Applications in Drones," *Drones*, vol. 7, no. 7, p. 427, 2023, doi: 10.3390/drones7070427.

[26] B. Chen, L. Cao, C. Chen, Y. Chen, and Y. Yue, "A comprehensive survey on the chicken swarm optimization algorithm and its applications: state-of-the-art and research challenges," *Artif. Intell. Rev.*, vol. 57, no. 7, p. 170, 2024, doi: 10.1007/s10462-024-10786-3.

[27] T. R. Alsenani, S. I. Ayon, S. M. Yousuf, F. B. K. Anik, and M. E. S. Chowdhury, "Intelligent feature selection model based on particle swarm optimization to detect phishing websites," *Multimed. Tools Appl.*, vol. 82, no. 29, pp. 44943–44975, 2023, doi: 10.1007/s11042-023-15399-6.

[28] P. Pathak and A. K. Shrivas, "Development of Proposed Model Using Random Forest with Optimization Technique for Classification of Phishing Website," *SN Comput. Sci.*, vol. 5, no. 8, p. 1059, 2024, doi: 10.1007/s42979-024-03388-x.

[29] N. K. Y. Gurukala and D. K. Verma, "Feature Selection Using Particle Swarm Optimization and Ensemble-Based Machine Learning Models for Ransomware Detection," *SN Comput. Sci.*, vol. 5, no. 8, p. 1093, 2024, doi: 10.1007/s42979-024-03454-4.

[30] W. Hu, Q. Cao, M. Darbandi, and N. Jafari Navimipour, "A deep analysis of nature-inspired and meta-heuristic algorithms for designing intrusion detection systems in cloud/edge and IoT: state-of-the-art techniques, challenges, and future directions," *Cluster Comput.*, vol. 27, no. 7, pp. 8789–8815, 2024, doi: 10.1007/s10586-024-04385-8.