International Journal of Computer Vision and Signal Processing, 15(1), 120-129(2025) ORIGINAL ARTICLE

Phishing Detection Using Gradient-Weighted Ensemble with Hybrid Sampling and Weighted Ensemble Feature Selection

Mrinal Basak Shuvo[1], Arjon Talukder[1], Sadia Islam Neela[1], Prakriti Paramarthi Roy[1], Tangina Sultana[2], Md. Delowar Hossain[1], Md. Arshad Ali[1], Eui-Nam Huh[3]

[1] *Department of Computer Science and Engineering, Hajee Mohammad Danesh Science & Technology University, Dinajpur 5200, Bangladesh.*

[2] *Department of Electronics and Communication Engineering, Hajee Mohammad Danesh Science & Technology University, Dinajpur 5200, Bangladesh.*

[3] *Department of Computer Science and Engineering, Kyung Hee University, Yongin-si 17104, Korea*

Abstract

Phishing attacks lead to a significant cybersecurity threat, where users get tricked with URLs and attackers extract sensitive information. As a result, the attackers gain an advantage and subsequently inflict significant damage. Hence, we have proposed a Machine Learning pipeline that facilitates the reduction of impact caused by phishing attacks. This study explores a competent architecture using various ML classifiers, including Random Forest, XGBoost, and LightGBM. We have used hybrid sampling involving SMOTE, TomekLinks and ADASYN, termed as Triad Sampling Fusion (TSF). For feature selection, we have used a weighted ensemble technique combined with Boruta, RFE, and Elastic Net, referred to as Boruta-RFE-ElasticNet Feature Selector (BREN-FS). For the enhancement of the performance of our model, we have performed GridSearchCV for Hyperparameter Tuning and developed a novel custom-weighted ensemble approach, termed as the Gradient Unified Weighted Ensemble (GUWE), that combines predictions from multiple models using gradient descent to optimize weights dynamically in order to increase classification performance. For training and evaluation, we have used the Mendeley Phishing Dataset. Our used dataset consists of 88,647 instances along with 111 features. Existing studies retain a relatively large feature subset, which increases computational complexity and overfitting risks. By applying TSF, BREN-FS, and comprehensive evaluation approaches such as GUWE, our study achieved an accuracy of 97. 52% that improves the effectiveness of phishing detection, significantly contributing to cybersecurity and reducing online threats.

Keywords: phishing attack, url, cybersecurity, GUWE, TSF, BREN-FS

IJCVSP

ISSN: 2186-1390 (Online)
http://cennser.org/IJCVSP

Article History:
Received: 10/4/2025
Revised: 11/7/2025
Accepted: 1/11/2025
Published Online: 23/11/2025

1. INTRODUCTION

As the Internet becomes an integral part of everyday life, people rely on it for communication, financial transactions, and accessing information. However, with this growing dependence, cyber threats such as phishing websites have also increased, compromising user security and privacy [1]. Phishing websites are a significant cybersecurity concern, resulting in financial fraud, identity theft, and data breaches on a global scale. Cybercriminals use deceptive strategies to replicate legitimate websites, tricking users into disclosing sensitive details like login credentials, credit card numbers, and personal information. These fraudulent sites closely resemble the legitimate ones,

Email addresses: mrinalbasak23@gmail.com (Mrinal Basak Shuvo[1]), arjontalukder100@gmail.com (Arjon Talukder[1]), sadianeela968@gmail.com (Sadia Islam Neela[1]), prakritiparamarthiroy2222@gmail.com (Prakriti Paramarthi Roy[1]), tangina@hstu.ac.bd (Tangina Sultana[2]), delowar@khu.ac.kr (Md. Delowar Hossain[1]), arshad@hstu.ac.bd (Md. Arshad Ali[1]), johnhuh@khu.ac.kr (Eui-Nam Huh[3])

creating significant challenges for users to differentiate between them [2]. Conventional detection methods, including blacklists and rule-based approaches, often fail to keep pace with the constantly evolving nature of phishing attacks [3]. As a result, Machine Learning (ML) techniques provide a robust and scalable approach by analyzing URL-based features to enhance the detection of website phishing.

According to cybersecurity reports, phishing attacks have increased significantly, with thousands of new phishing URLs emerging daily. Reports indicate that around 15 billion spam emails are sent daily, and in 2021 [4], 83% of the organizations faced phishing attacks. Furthermore, studies show that more than one million phishing incidents were recorded worldwide in the first quarter of this year, marking it as one of the most severe periods for phishing attacks. Relying on manually updated blacklists is insufficient, as new phishing websites are often short-lived and generated dynamically. As a result, ML-based models that analyze URL structure, domain-related features, and network attributes can provide real-time and scalable phishing detection. However, high detection accuracy demands robust classifiers, best feature selection, and a strong pipeline that maximizes the classification performance. Aligned with current trends in information security, most notably the increased reliance on AI-driven threat detection, real-time monitoring, zero-trust architecture, and adaptive security models, our machine learning-based phishing detection system aligns with the emerging paradigm shift toward proactive, intelligent, and automated cyber defence systems. This demonstrates the technical validity of our approach to address the evolving and multifaceted nature of modern-day phishing attacks. Previous studies have emphasized feature selection to improve phishing website detection; however, they retain a significantly larger subset of characteristics, 53 out of 111 [5] and 51 [6], highlighting the need for further optimization. Despite feature reduction, their models still incorporate a relatively large feature set, potentially increasing computational complexity and overfitting risks. In comparison, our approach further refines feature selection, leveraging only 35 key features, which is 31.52% of the total features, while maintaining high accuracy. This demonstrates a more efficient and optimized approach, which develops an advanced weighted ensemble feature selection technique that improves classification performance while minimizing redundancy.

Our aim in this study is to optimize the detection of phishing websites based on how various data preprocessing, feature selection methods, and bespoke ensemble models affect model classification performance. We begin with data cleaning and handle missing data using median imputation. Data quality is also ensured by removing features with zero variance. For classification, we implement Random Forest Classifier (RFC), XGBoost Classifier (XGBC), and LightGBM (LGBM), evaluating their performance using standard metrics such as accuracy, precision, recall, and F1-score. To address the class imbalance, we apply a hybrid sampling technique, including SMOTE, Tomek-Links, and ADASYN, termed Traid Sampling Fusion (TSF), ensuring a balanced dataset for training. Furthermore, we employ a weighted ensemble feature selection method involving Boruta, Recursive Feature Elimination (RFE), and ElasticNet, named Boruta-RFE-ElasticNet Feature Selector-(BREN-FS), to identify the most relevant features for phishing detection. To enhance model performance, GridSearch-CV is utilized for hyperparameter tuning, optimizing the configuration of each classifier. We further improve prediction accuracy by designing a custom weighted ensemble approach, which we coined as Gradient Unified Weighted Ensemble (GUWE), that dynamically combines predictions from various models using gradient descent to optimize weights to increase classification performance.

By integrating TSF, BREN-FS, and modified ML classifiers, this study aims to develop a highly effective and transparent model for identifying phishing websites. The key contributions of our research are as follows:

- We propose a ML-based prediction pipeline that significantly enhances the accuracy of phishing website classification by incorporating a TSF approach.
- Our study introduces a hybrid weighted ensemble feature selection approach BREN-FS, that combines three different feature selection techniques.
- We develop a novel custom weighted ensemble approach named GUWE using gradient descent and L2 regularization to optimize the weights of these models based on a validation set's log loss. GUWE outperforms individual ML classifiers in a binary classification setting.
- We discreetly select RFC, XGBC, and LGBM to design the GUWE model as the base classifier because of their complementary strengths. The selection of these models improves the ability of the ensemble to detect phishing.
- We apply hyperparameter tuning using GridSearch-CV to advance the base models. After applying fine-tuning to each model, the ensemble's performance is maximized, and prediction accuracy is improved.

The paper is organized in the following manner. We describe an overview of the existing approaches in phishing website detection and highlight their methodology and limitations in Section 2. We describe the architecture of our proposed model, including algorithms of the TSF, BREN-FS, Gradient Unified Weighted Ensemble (GUWE), and in Section 3. Then, Section 4 explains the result of our experiment in detail, displaying performance comparison through the different tables and visualizations such as AUC-ROC and Precision-Recall curves. Finally, we conclude our analysis in Section 5 by summarizing our findings.

2. Literature Review

As we advance daily, it has become crucial for us to protect ourselves from phishing websites. This has become a serious issue worldwide, as we can't go a single day without using technology. So, we have proposed a model that will help us detect phishing websites more accurately compared to some studies.

In the study [6], they have introduced a feature selection approach, which consists of filter and wrapping methods used in 2 phases. Artificial Neural Network(ANN), XGBC, and RFC are used to perform 10-fold cross-validation on Data-I, where XGBoost outran the performance with an accuracy of 96.08% and the XGBoost model performed with an accuracy of 97.29% on Data-II.

In study [5], they have presented a method for detecting phishing attacks using ML, consisting of support vector machine (SVM), logistic regression (LR), K-Nearest Neighbors (KNN), Decision Tree (DT), Gradient Boosting (GB), LGBM, Extreme Gradient Boosting, Categorical Boosting, Neural Networks, Multilayer Perceptrons, and online ML algorithms. By using feature selection methodology, they achieved the highest accuracy with the DT classifier on the existing dataset, about 98.96%, and 93.67% in the new dataset.

In the study by Onih [7], a robust phishing detection system was developed using ML algorithms such as KNN, ANN, and RFC. After optimizing their hyperparameters with GridSearchCV, the RFC outperformed the rest with an accuracy of 99.78% and was integrated into a Django-based web application for real-time phishing detection.

In another study [8], an ML approach was proposed using permutation importance based on feature selection techniques and Deep Learning. Information Gain, Chi-Square, and Fisher's Score were used to select the best features. Deep Neural Network (DNN), Wide and Deep, and TabNet classifiers were used and optimized with hyperparameter-sensitive grid search. The model achieved an accuracy of 94.46%. Furthermore, they have introduced a novel anti-phishing score in their study. This metric improved the model's capability to detect phishing attacks with greater precision.

In a similar study in [9], they proposed an optimized stacking ensemble method using a Genetic Algorithm (GA) for the tuning of the parameters of several ensemble ML methods, including LGBM, GB, RFC, AdaBoost, Bagging, and XGBC. With these techniques, the model achieved accuracies of 97.16%, 98.58%, and 97.39% across three datasets.

For comparison, Table 1 is presented, summarizing the methodologies, performance metrics, limitations, and datasets utilized in comparable studies.

Table 1: Analysis of Prior Research Work

Ref.	Dataset	Best Methodoloy	Performance (Accuracy)	Limitation
[6]	Grega Vrbančič (Mendeley) [10]	XGBoost with hybrid feature selection	DATA-I 96.08%, DATA-II 97.29%	Missing real-time detection challenges.
[5]	UCI repository (2015)[11]	DT with Sequential Feature Selection	98.96%, 93.67% (new dataset)	The model will degrade in new dataset.
[7]	Ariyadasa et al. (2021)[12] and UNB (2016)[13]	RFC with manual and automated Feature Selection	99.78%	Potentially limiting the model's effectiveness against emerging threats.
[8]	Grega Vrbančič (Mendeley) [10]	FNN + *Permutation* importance based on Feature Selection + 10foldCV + Hyperparameter GridSearch	94.46%	Dataset diversity, adversarial attack vulnerability.
[9]	UCI (2019) [14],Grega Vrbančič[10], Mendeley Data (2018)[15]	Stacking Ensemble method + 10-foldCV	97.16%, 98.58%, 97.39%	This study might affect the robustness in real-world scenarios.

3. Proposed Methodology

In our proposed model, we utilized Vrbancic, Grega (2020), "Phishing Websites Dataset," Mendeley Data for Preprocessing. We used TSF for sampling, consisting of SMOTE, TomekLinks, and ADASYN for the imbalance, and we applied median imputation to handle missing values in the dataset. We adopted a hybrid weighted ensemble BREN-FS for feature selection, involving Boruta, RFE, and Elastic Net to extract the most relevant features. Traditional ML classifiers, including RFC, XGBC, and LGBM, were trained on the preprocessed dataset, with hyperparameter tuning performed using GridSearchCV. Additionally, we integrated a custom weighted ensemble model - (GUWE) that combines the strengths of RFC, XGBC and

LGBM using gradient descent and L2 regularization. The pipeline of our proposed model is given in Figure 1.

3.1. Data Preprocessing

For the data preprocessing, we used a publicly available dataset [10], which is a URL-based dataset. This algorithm preprocesses the data of 88,647 rows and 112 columns for phishing website detection by beginning with cleaning the data and handling missing values, removing columns with over 80% missing entries (replaced as NaN from -1 or non-numeric values), and imputing remaining gaps with medians, while also dropping features with zero variance.

The dataset was then split into 70% training and 30% temporary sets (further divided into validation and test) to preserve class distribution.

In an effort to balance the class, the resampling technique TSF uses SMOTE (minority to 75 % majority), helping the model learn broader patterns. TomekLinks then removes overlapping or noisy instances that lie near class boundaries, and ADASYN (full balance) adaptively adds synthetic samples in regions where the minority class is harder to learn, further enhancing model generalization. It performs better than single oversampling such as SMOTE, ADASYN and undersampling such as TomekLinks. Descriptive processes are given in Algorithm 1.

3.2. Feature Engineering

In this paper, we combined three feature selection methods such as Boruta, RFE, and Elastic Net to form the BREN-FS ensemble feature selection approach. Boruta helps us identify the most important features by comparing them to random duplicates, RFE eliminates less relevant features to make the model more efficient, and Elastic Net balances feature selection with regularization to handle correlated features. These techniques are integrated into a hybrid model, which enhances the accuracy and interpretability of our phishing website detection system.

Algorithm 1 Data Preprocessing and Triad Sampling Fusion (TSF)

Input: Dataset D
Output: Preprocessed and resampled training data $X_{resampled}$, $y_{resampled}$, test data X_{test}, y_{test}, validation data X_{val}, y_{val}
Step 1: Data Cleaning and Missing Values Handling
if $\texttt{NaN}(D) = \emptyset$ **then**
  Proceed
end if
Replace -1 and non-numeric values in D with `NaN`
for each column $c \in D$ **do**
  if $\%\texttt{NaN}_c > 80\%$ **then**
    Remove c
  end if
end for
Initialize `imputer` $\leftarrow$ SimpleImputer(strategy = "median")
for each numerical $c \in D$ **do**
  $c \leftarrow$ `imputer.fit_transform`(c)
end for
for each $c \in D \setminus \{\text{target}\}$ **do**
  if $\text{Var}(c) = 0$ **then**
    Remove c
  end if
end for
Step 2: Data Splitting
Split D into features X (all columns except target) and target y (target column)
X_{train}, y_{train}, X_{temp}, $y_{temp} \leftarrow$ Split (X, y) (test_size = 0.3, stratify = y) ▷ 70% training, 30% temporary
X_{val}, y_{val}, X_{test}, $y_{test} \leftarrow$ Split (X_{temp}, y_{temp}) (test_size = 0.5, stratify = y_{temp}) ▷ validation, test
Step 3: TSF
Initialize `smote` $\leftarrow$ SMOTE(sampling_strategy = 0.75)
X_{smote}, $y_{smote} \leftarrow$ Resample X_{train}, y_{train} using `smote` ▷ minority class to 75% of majority
Initialize `tomek` $\leftarrow$ TomekLinks()
X_{smote_tomek}, $y_{smote\_tomek} \leftarrow$ Apply `tomek` on X_{smote}, y_{smote} ▷ remove noisy samples
Initialize `adasyn` $\leftarrow$ ADASYN(sampling_strategy = 1.0)
$X_{resampled}$, $y_{resampled} \leftarrow$ Resample X_{smote_tomek}, y_{smote_tomek} using `adasyn` ▷ to fully balance classes
Step 4: Output
Return $X_{resampled}$, $y_{resampled}$, X_{test}, y_{test}, X_{val}, y_{val}

Boruta : Boruta is a comprehensive feature selection technique that functions as a wrapper for classification algorithms, primarily leveraging RFC. It streamlines feature selection by automatically identifying thresholds and extracting the most significant features from the dataset. The concept behind Boruta is both intriguing and straightforward. Random duplicates (known as shadow features) are generated for every feature in the original dataset, and

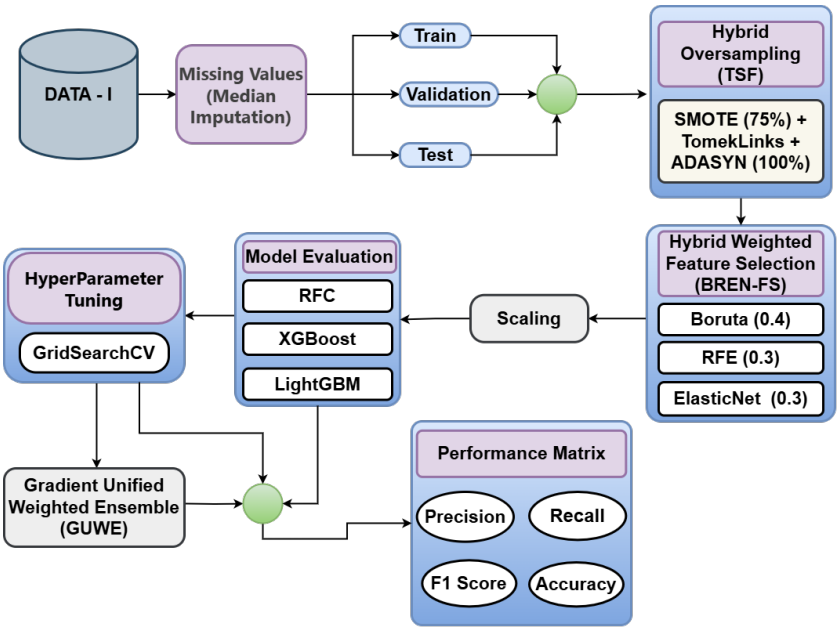

Figure 1: Architecture of the proposed methodology

classifiers are trained using this expanded dataset.

$$Z_i = \frac{\text{mean importance}_i - \text{max shadow importance}}{\text{std dev of importance}},$$

where Z_i exceeding a threshold (e.g., from binomial distribution) flags feature i as significant.

RFE: Recursive Feature Elimination (RFE) is a wrapper-based feature selection technique deployed as a systematic feature selection strategy within the feature engineering framework to refine the predictive capabilities of the classification models. It systematically trains an estimator, such as a Random Forest or gradient boosting model, on the full 50-feature dataset. Unlike filter-based selection methods, which assign scores to individual features and retain those with the highest or lowest values, RFE systematically eliminates less essential features to enhance model performance.

$$R_i = \text{iteration of removal},$$

where features with the lowest R_i (highest importance) are selected up to k features.

Elastic Net : Elastic Net is an ML technique used for feature selection and regression. It combines the strengths of both Lasso (L1) and Ridge (L2) regression to improve model performance. Lasso helps in selecting the most important features by shrinking less relevant ones to zero, while Ridge prevents overfitting by distributing the penalty across all features. This makes Elastic Net particularly useful when dealing with datasets with many correlated features. Adjusting two tuning parameters, alpha and lambda, effectively balances feature selection and regularization to enhance model accuracy.

$$\min\left(\frac{1}{2N}\sum(y - Xw)^2 + \lambda\left(\alpha\sum|w_i| + \frac{1-\alpha}{2}\sum w_i^2\right)\right.$$

, where features with non-zero w_i are retained.

The Boruta-RFE-ElasticNet Feature Selector (BREN-FS) integrates Boruta's robust feature importance, RFE's iterative elimination, and Elastic Net's regularization, a combination of filter, wrapper and embedded methods to select an optimal feature subset from the 111 features, effectively balancing the bias-variance trade-off by reducing overfitting and improving model generalization. The process description of BREN-FS is given in Algorithm 2.

Algorithm 2 Boruta-RFE-ElasticNet Feature selector (BREN-FS)

Input: Training data: $X_{resampled}$, $y_{resampled}$, test data: X_{test}, validation data: X_{val}
Output: Subset of selected features
Step 1: Boruta Feature Selection
Initialize $RFC \leftarrow RandomForestClassifier(n\_jobs = -1, class\_weight =' balanced', max\_depth = 5)$
Initialize $Boruta \leftarrow BorutaPy(RFC, n\_estimators =' auto', max\_iter = 30)$
Fit $Boruta$ on $X_{resampled}$, $y_{resampled}$
$boruta\_features \leftarrow$ features where $Boruta.support\_=True$
Step 2: Elastic Net Feature Selection
Initialize $EN \leftarrow ElasticNetCV(cv = 5, l1\_ratio = [0.1, 0.5, 0.7, 0.9, 0.95, 0.99, 1])$
Fit EN on $X_{resampled}$, $y_{resampled}$
$elastic\_net\_features \leftarrow$ features where $EN.coef\_ \neq 0$
Step 3: RFE Feature Selection
Initialize $RFE \leftarrow RFE(RandomForestClassifier(), n\_features\_to\_select = 50)$
Fit RFE on $X_{resampled}$, $y_{resampled}$
$rfe\_features \leftarrow$ features where $RFE.support\_=True$
Step 4: Weighted Combination
Define weights: $w_{boruta} = 0.4$, $w_{elastic} = 0.3$, $w_{rfe} = 0.3$
$boruta\_scores \leftarrow$ [1 if feature in $boruta\_features$ else 0]
$elastic\_net\_scores \leftarrow$ [1 if feature in $elastic\_net\_features$ else 0]
$rfe\_scores \leftarrow$ [1 if feature in $rfe\_features$ else 0]
$weighted\_scores \leftarrow w_{boruta} \cdot boruta\_scores + w_{elastic} \cdot elastic\_net\_scores + w_{rfe} \cdot rfe\_scores$
$selected\_features \leftarrow$ features ($weighted\_scores > 0.6$)
Step 5: Output
Return $X_{train}[sel\_fea]$, $X_{test}[sel\_fea]$, $X_{val}[sel\_fea]$ ▷ sel_fea=Selected Features

3.3. ML Classifiers

In this paper, we used three ML classifiers RFC, XGBC, and LGBM to enhance prediction accuracy. RFC combines decision trees to improve reliability, XGBC builds on previous models to correct mistakes, and LGBM handles large datasets quickly. Our unique approach, **GUWE**, combines these models by optimizing their weights to make the most accurate predictions, especially when dealing with imbalanced datasets. This combination of models helps us get more reliable classification results.

RFC : RFC is a highly effective ML classifier that uses an ensemble of decision trees to achieve impressive classification results. By incorporating feature randomness through the random subspace method, RF creates diverse decision trees, reducing correlation and enhancing reliability. Unlike traditional decision trees that evaluate all feature splits, Random Forest focuses on a limited subset of features, improving efficiency and accuracy. Its simplicity and adaptability make it a preferred choice for classification and regression tasks, consistently delivering dependable outcomes. With predictions combined through majority voting for classification or averaging for regression, Random Forest effectively turns data challenges into successful solutions.

$$P_{RF}(y) = \frac{1}{T}\sum_{t=1}^{T} P_t(y),$$

Where $P_t(y)$ is the probability of class y from the t-th tree, $P_{RF}(y)$ is the averaged probability across all T trees, and T is the total number of trees in the Random Forest.

XGBC : XGBC is an admiringly optimized gradient boosting classifier for efficient and accurate predictive modeling. Combining decision trees one after another and using the previous one, each tree is designed to address the mistake, leading to an overall improvement. It integrates regularization strategies like L1 and L2 penalties, ensuring model stability. Known for its computational efficiency, it swiftly processes large datasets and handles missing values effectively. It also offers valuable insights by evaluating feature importance, making it a powerful tool for various predictive tasks. If we refine model performance through gradient-based learning, this core mechanism involves optimizing a loss function (e.g., log loss).

$$\hat{y} = \sum_{k=1}^{K} f_k(x),$$

where $\hat{y}$ is the final prediction made by aggregating the contributions of all trees,$f_k(x)$ is the output of the kth tree, and K is the total trees, with loss $\mathrm{L} = \sum l(y, \hat{y}) + \sum \Omega(f_k)$.

LGBM : LGBM, short for Light Gradient Boosting Machine, is an advanced, open-source gradient boosting platform originally crafted by Microsoft. This framework is built for high performance and excels efficiently on large and complex datasets. LGBM enhances predictive accuracy by building decision trees leafwise and selecting optimal splits to reduce loss. Its notable features are the capability for parallel and distributed computing, efficiency in managing large data volumes, and the application of advanced gradient-boosting techniques like histogram-based methods, which help reduce memory consumption and faster processing. Widely appreciated for its speed and low memory footprint, LGBM is extensively used in various ML applications such as classification and ranking.

$$\hat{y} = \sum_{k=1}^{K} w_k \cdot \mathrm{leaf}(x)$$

where $\hat{y}$ is the ground truth used to evaluate and refine the model's predictions. w_k is the weight of leaf nodes,

optimized via gradient descent with a loss function like $L = \sum l(y, \hat{y})$.

Algorithm 3 Gradient Unified Weighted Ensemble (GUWE)

Input:
Validation probabilities $\mathbf{P}_{val} = (P_{rf}, P_{xgb}, P_{lgbm})$
Test probabilities $\mathbf{P}_{test} = (P'_{rf}, P'_{xgb}, P'_{lgbm})$
Validation labels y_{val}, Test labels y_{test}
Iterations T, Learning rate η, Regularization λ, Minimum weight w_{min}
Output:
Optimized weights $\mathbf{w}$, Ensemble test predictions $\hat{y}_{test}$

$\mathbf{w} \leftarrow [1/3, 1/3, 1/3]$ ▷ Initialize equal weights
for $t = 1$ to T **do**
  $P_{ens,val} \leftarrow w_1 \cdot P_{rf} + w_2 \cdot P_{xgb} + w_3 \cdot P_{lgbm}$ ▷ Ensemble validation probabilities
  $L \leftarrow \text{log\_loss}(y_{val}, P_{ens,val}) + \lambda \cdot \sum_{i=1}^{3} w_i^2$ ▷ Loss with regularization
  $g_1 \leftarrow \frac{1}{N} \sum_{i=1}^{N} (P_{ens,val,i} - y_{val,i}) \cdot P_{rf,i} + 2\lambda w_1$ ▷ Gradient for RF
  $g_2 \leftarrow \frac{1}{N} \sum_{i=1}^{N} (P_{ens,val,i} - y_{val,i}) \cdot P_{xgb,i} + 2\lambda w_2$ ▷ Gradient for XGB
  $g_3 \leftarrow \frac{1}{N} \sum_{i=1}^{N} (P_{ens,val,i} - y_{val,i}) \cdot P_{lgbm,i} + 2\lambda w_3$ ▷ Gradient for LGBM
  $\mathbf{g} \leftarrow [g_1, g_2, g_3]$
  $\mathbf{w} \leftarrow \mathbf{w} - \eta \cdot \mathbf{g}$ ▷ Update weights
  $\mathbf{w} \leftarrow \max(\mathbf{w}, w_{min})$ ▷ Enforce minimum weight
  $\mathbf{w} \leftarrow \mathbf{w} / \sum_{i=1}^{3} w_i$ ▷ Normalize weights
end for
$P_{ens,test} \leftarrow w_1 \cdot P'_{rf} + w_2 \cdot P'_{xgb} + w_3 \cdot P'_{lgbm}$ ▷ Ensemble test probabilities
$\tau_{best} \leftarrow 0.5$, $acc_{best} \leftarrow 0$ ▷ Initialize threshold optimization
for $\tau \in \{0.41, 0.42, \ldots, 0.60\}$ **do**
  $\hat{y}_{val} \leftarrow (P_{ens,val} \geq \tau)$
  $acc \leftarrow \text{accuracy}(y_{val}, \hat{y}_{val})$
  if $acc > acc_{best}$ **then**
    $acc_{best} \leftarrow acc$
    $\tau_{best} \leftarrow \tau$
  end if
end for
$\hat{y}_{test} \leftarrow (P_{ens,test} \geq \tau_{best})$ ▷ Final test predictions
return $\mathbf{w}, \hat{y}_{test}$

GUWE : The GUWE algorithm is a novel and custom weighted ensemble approach. This approach is theoretically grounded in ensemble learning's emphasis on diversity, as combining uncorrelated models minimizes variance while maintaining low bias, resulting in a combination of distinct ML models like RFC, XGBC, and LGBM to improve overall performance and dynamically assigns weights to their output. The algorithm optimizes these weights based on the log loss of a validation set, and to ensure the contribution of models efficiently, it enforces a minimum weight constraint. It also follows L2 regularization to reduce overfitting and ensure proper weight distribution.

$$L = \text{log\_loss}(y_{\text{val}}, \text{en\_val}) + \lambda \sum_{m \in \{\text{RFC,XGBC,LGBM}\}} w_m^2$$

where L is the total loss, y_{val} denotes the validation labels, en_val is the ensemble probability on the validation set, λ is the regularization parameter, and w_m represents the weight of model m. Algorithm 3 outlines the detailed process.

4. Result and Discussion

In this section, We provide the results and analysis of our proposed study. Evaluation tables and figures reveal the strengths and weaknesses of the ML classifiers. To evaluate our proposed system, we deployed a prototype written in Python on a Lenovo IdeaPad Slim 3 laptop running Windows. The computer has an AMD Ryzen 7 7730U processor (8-core/16-thread, up to 4.5 GHz), 8GB DDR4 RAM, and a 512GB SSD.

The study utilizes the Phishing Websites Dataset, provided by Grega Vrbanˇciˇc on September 24, 2020 [10]. The dataset employed by this study has 88,647 instances of webpages, of which 58,000 are genuine, and 30,647 are phishing pages containing 111 features that can be used to detect phishing. Although the dataset is class-imbalanced and has more genuine pages than phishing pages, it contains a sufficient phishing instances to train an effective model. Even if the dataset cannot show the latest phishing techniques or view all regions and attack types, it is valuable in terms of phishing website detection. Moreover, with effective feature selection, the danger of noisy or irrelevant data can be minimized. All these factors must be considered at the training and interpretation stage, but in general, the dataset remains a good foundation for phishing detection research. In anti-phishing systems operating in real-time, models must ensure high predictive accuracy and working efficiency. Our model is suitable for use in such scenarios, and BREN-FS reduces the dimensionality of the characteristics from 111 to 35, reducing the computational complexity by 68%. The GUWE ensemble is highly accurate with minimal runtime overhead and is modularly designed for parallel, scalable processing. These features allow efficient deployment in high-throughput cybersecurity environments.

4.1. Evaluation of the ML models

This section presents a detailed evaluation of the ML models, including RFC, XGBC, and LGBM, through performance metrics (Accuracy, Precision, Recall, F1-Score, AUC-ROC), hyperparameter tuning, and ensemble analysis.

Table 2: PERFORMANCE OF THE CLASSIFIER MODELS

Algorithm	Class	AUC-ROC	Precision	Recall	F1 Score	Avg-Acc%
RFC	0.0 1.0	**0.9951**	**0.9819** **0.9497**	**0.9733** **0.9657**	**0.9776** **0.9576**	**97.07**
XGBC	0.0 1.0	0.9935	0.9804 0.9331	0.9639 0.9631	0.9721 0.9479	96.37
LGBM	0.0 1.0	0.9948	0.9805 0.9425	0.9693 0.9631	0.9749 0.9527	96.72

Table 2 shows the summarized performance of our evaluation models. With the use of Various classifiers, including RFC, XGBC, and LGBM, we have achieved an accuracy of 97.07%, the highest in balanced data with RFC, with decent class separation.

Table 3: TUNED HYPERPARAMETERS FOR DIFFERENT ML MODELS

Models	Best Estimators
RFC	'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100
XGBC	'learning_rate': 0.1, 'max_depth': 6, 'n_estimators': 200, 'subsample': 0.7
LGBM	'boosting_type': 'gbdt', 'learning_rate': 0.1, 'n_estimators': 200, 'num_leaves': 50

Table 3 hyperparameters were optimized with GridSearchCV for optimum performance of each model. For RFC, full depth of the tree and low thresholds for splitting allowed for learning intricate patterns. XGBC used balanced depth ($max\_depth = 6$) and a moderate learning rate (0.1), and subsampling (0.7) prevented overfitting. LGBM also used a controlled *num_leaves* and a consistent learning rate. These optimized parameters improved precision and model generality, especially when included in our ensemble approach (GUWE).

Table 4: PERFORMANCE OF THE CLASSIFIERS AFTER HYPERPARAMETER TUNING

Algorithm	Class	AUC-ROC	Precision	Recall	F1 Score	Avg-Acc%
RFC	0.0 1.0	0.9951	0.9817 0.9494	0.9731 0.9653	0.9774 0.9573	97.04
XGBC	0.0 1.0	0.9955	0.9816 0.9524	0.9748 0.9650	0.9783 0.9587	97.14
LGBM	0.0 1.0	**0.9962**	**0.9835** **0.9567**	**0.9771** **0.9686**	**0.9803** **0.9626**	**97.42**

Following hyperparameter tuning through GridSearchCV, our results for Table 4, reflect an achieved accuracy level of 97.42%, of which LGBM was the best among all models. This is primarily owed to the optimal adjustment of fundamental hyperparameters such as learning rate, estimators, and max depth, which greatly increased the generalization capacity of the model while enabling accurate predictions.

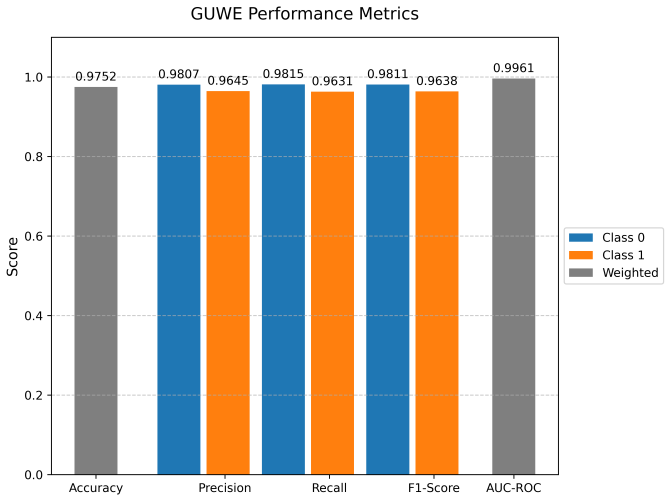

Figure 2: GUWE Performance Metrics

The performance metrics Figure 2 show a visual representation of the capabilities of the GUWE model. In these metrics, we can see the performance of class 1 and class 0. Here we have achieved 97.52% average accuracy and AUC-ROC of 99.61% and in the context of class 0, the performance is better compared to class 1, achieving precision, recall, and f1-score respectively 98.07%, 98.15%, 98.11% and for class 0, achieving precision, recall, and f1-score respectively 96.45%, 96.31%, 96.38% better than other classifier models. The consistency and strong performance of the ensemble in all evaluation metrics reflect its potential applicability.

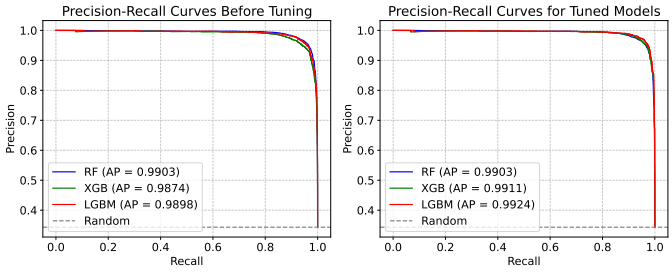

Figure 3: Precision-Recall Curves for Untuned and Tuned Model

The illustration in Figure 3 including the Precision-Recall curves compares model performance before and after tuning. After tuning, XGBC and LGBM show improved average precision (AP: 0.9911 and 0.9924, respectively), while Random Forest remains stable (AP: 0.9903). All models maintain high precision and recall, indicating

strong classification performance with minimal false positives and false negatives.

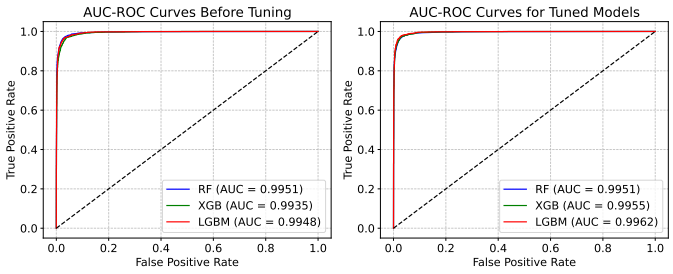

Figure 4: AUC-ROC Curves for Untuned and Tuned Model

This comparison of the AUC-ROC curve shows the performance of the RFC, XGBC, and LGBM models before and after hyperparameter tuning in Figure 4. XGBC and LGBM improved after tuning, achieving AUCs of 0.9955 and 0.9962 respectively, while RFC remained steady at 0.9951. All models demonstrate excellent classification capability with AUCs near 1.

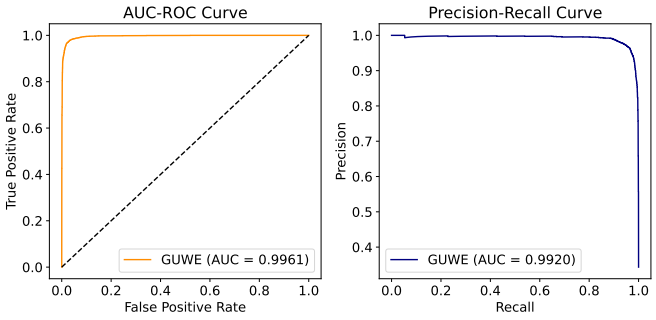

Figure 5: Auc-Roc and Precision-Recall curve of GUWE

The GUWE model shows excellent performance with an AUC-ROC of 0.9961 and PR AUC of 0.9920 in Figure 5. The AUC-ROC curve indicates strong class separation, while the PR curve reflects high precision across recall levels. Overall, the model is highly accurate and reliable for classification tasks.

4.2. Comparison with Previous Studies

We compared the results of our ensemble model with previous studies that used "Phishing Websites Dataset" and Mendeley Data. Grega Vrbančič et al.[16] used a DNN with firefly optimization, achieving 90.17% accuracy with 111 features. Pankaj Bhowmik et al.[6] used XGBC with hybrid feature selection, reaching 96.08% accuracy with 51 features. Berende P. et al.[5] combined multiple models (RFC, XGBC, MLP, GB, DT) and achieved 95.25% accuracy using 53 features. Our proposed model combines TSF, BREN-FS, and GUWE (Gradient Unified Weighted Ensemble), using only 35 features which is 31.52% of the initial features. It outperforms previous approaches, achieving the highest accuracy of 97.52%. The outcome is depicted in Table 5.

Table 5: PERFORMANCE COMPARISON BETWEEN PRIOR STUDIES AND THE PROPOSED ENSEMBLE-BASED METHOD

Ref.	Best Approach	Features	Results (Accuracy)
Grega Vrbancic et al.[16]	DNN, optimized with firefly algorithm	111	90.17%
Pankaj Bhowmik et al.[6]	XGBC, with hybrid feature selection	51	96.08%
Berende, P. et al.[5]	RFC+XGBC+MLP (Multilayer Perceptron)+GBDT (Gradient Boosted Decision Tree with Neural Networks)	53	95.25%
Our Proposed Model	**TSF+ BREN-FS + GUWE(Gradient Unified Weighted Ensemble)**	**35**	**97.52%**

5. Conclusions and Future Work

Phishing is an online fraud in which hackers tempt users to get personal information such as passwords, credit cards, or bank account data. A phishing website is a domain similar in name and appearance to an official website. Here, we introduced URL-based phishing website detection using an advanced ML approach, reducing limitations in traditional methods through a robust pipeline of data preprocessing, feature selection, and classification. By utilizing median imputation and TSF, we effectively handled missing data and class imbalance in the "Phishing Websites Dataset." Integrating Boruta, RFE, and Elastic Net as a weighted ensemble, the BREN-FS reduced the feature set to 35 which is 31.53 % of the initial features while preserving predictive power. RFC, XGBC, and LGBM are implemented and enhanced by GridSearchCV for hyperparameter tuning, giving accuracy exceeding 97% and showing well-defined individual performance. However, the proposed Gradient Unified Weighted Ensemble (GUWE) surpassed these classifiers, achieving an accuracy of 97.52% and an AUC-ROC of 99.61%, outperforming prior studies on the same dataset. By eliminating marginal features, reducing computational complexity, and evading the risk of overtraining, BREN-FS improved the efficiency and interpretability of models. TSF helped in enhancing fairness and stability in models by managing class imbalance effectively. GUWE is an encouraging prospect for further universal categorization

tasks due to its flexibility and scalability. Its use in multi-class problems and linkage to other advanced models can be explored in future research. All of these in combination enhanced accuracy in categorization and strengthened generalization across various phishing tactics. Our findings indicate that our model generates a practical outcome to protect users from online fraud and enhance cybersecurity.

To make our model even more applicable to real-world usage, further research could be extended to its modification to support multi-class classification so that it can make more nuanced phishing attempt detections. Further, real-time integration of the model into browser add-ons would allow proactive protection for users. Its modification to support multilingual datasets for phishing would make it more useful for application in various linguistic environments, making it available worldwide.

Acknowledgements

This research was partly supported by the MSIT (Ministry of Science and ICT), Korea, under the 1) Convergence security core talent training business support program (IITP-2024-RS-2023-00266615, 50%) and 2) ITRC (Information Technology Research Center) support program (IITP-2023-RS-2023-00258649, 50%) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation). Eui-Nam Huh and Tangina Sultana are the co-corresponding authors.

References

[1] Spiceworks, What is phishing? definition, types, and prevention best practicesAccessed: 2025-03-25.
URL https://www.spiceworks.com/it-security/vulnerability-management/articles/what-is-phishing-definition-types-and-prevention-best/

[2] C. P. S. T. Ltd., Phishing detection techniques.
URL https://www.checkpoint.com/cyber-hub/threat-prevention/what-is-phishing/phishing-detection-techniques/

[3] Sangfor Technologies, What is a phishing attack?Accessed: 2025-04-10.
URL https://www.sangfor.com/blog/cybersecurity/what-is-phishing-attack

[4] T. Networks, Fighting phishing attacks 101: Definition, methods, and moreAccessed: 2025-03-25.
URL https://www.timusnetworks.com/fighting-phishing-attacks-101-definition-methods-and

[5] A. Rachmani, Detecting phishing attacks by machine learning, Master's thesis, The Academic College of Tel-Aviv, Yaffo (n.d.).

[6] P. Bhowmik, M. Sohrawordi, U. A. M. E. Ali, P. C. Bhowmik, An empirical feature selection approach for phishing websites prediction with machine learning (2022) 173–188.

[7] V. Onih, Phishing detection using machine learning: A model development and integration, International Journal of Scientific and Management Research 07. doi:http://doi.org/10.37502/IJSMR.2024.7403.

[8] G. S. Nayak, B. Muniyal, M. C. Belavagi, Enhancing phishing detection: A machine learning approach with feature selection and deep learning models, IEEE Access 13 (2025) 33308–33320. doi:10.1109/ACCESS.2025.3543738.

[9] M. Al-Sarem, F. Saeed, Z. G. Al-Mekhlafi, B. A. Mohammed, T. Al-Hadhrami, M. T. Alshammari, A. Alreshidi, T. S. Alshammari, An optimized stacking ensemble model for phishing websites detection, Electronics 10 (11). doi:10.3390/electronics10111285.
URL https://www.mdpi.com/2079-9292/10/11/1285

[10] G. Vrbančič, Phishing websites dataset V1. doi:10.17632/72ptz43s9v.1.

[11] R. Mohammad, L. McCluskey, Phishing websites-doi:10.24432/C51W2X.

[12] S. Ariyadasa, S. Fernando, S. Fernando, Phishing websites dataset, Mendeley Data 1 (5) (2021) 10–17632.

[13] U. of New Brunswick, Url 2016 — datasets — research — canadian institute for cybersecurity — unb.
URL https://www.unb.ca/cic/datasets/url-2016.html

[14] D. Dua, C. Graff, Uci machine learning repository.
URL http://archive.ics.uci.edu/ml

[15] C. L. Tan, Phishing dataset for machine learning: Feature evaluation V1. doi:10.17632/h3cgnj8hft.1.

[16] G. Vrbančič, I. Fister, V. Podgorelec, Parameter setting for deep neural networks using swarm intelligence on phishing websites classification, International Journal on Artificial Intelligence Tools 28 (06) (2019) 1960008. doi:10.1142/S021821301960008X.
URL https://doi.org/10.1142/S021821301960008X