
Phishing Websites Detection based on Meta-heuristic Optimization and Ensemble Algorithms

Kayode S. Adewole^{1,2,*}, Moshood A. Hambali³, Yakub Saheed⁴, Victor E. Adeyemo⁵, Kamaldeen J. Muhammed⁶, Rafiu M. Isiaka⁷

¹Department of Computer Science and Media Technology, Malmö University, Malmö, Sweden

²Sustainable Digitalisation Research Centre, Malmö University, Malmö, Sweden

³Department of Computer Science, Federal University Wukari, Taraba, Nigeria

⁴School of IT & Computing, American University of Nigeria

⁵Department of Computer Science, Durham University, Durham, United Kingdom

⁶Department of Educational Technology, University of Ilorin, Ilorin, Nigeria

⁷Department of Computer Science, Kwara State University, Malete, Nigeria

*Corresponding author: kayode.adewole@mau.se

Abstract: Phishing attacks pose a significant threat to web- and mobile-based applications, aiming to compromise network security and hijack legitimate resources. Illegitimate websites, fake emails, and deceptive web/mobile solutions are used to collect user credentials and compromise accounts. Traditional blacklist-based methods for detecting phishing websites suffer from a lack of zero-day attack detection. To address this, machine learning (ML) models have been widely explored. However, meta-heuristic approach that covers studies on feature selection to create compact ML models for phishing detection has not been extensively studied. This paper fills that gap by analyzing ten meta-heuristic algorithms to identify discriminating features for detecting phishing websites. The study evaluates three ensemble-based classifiers - Random Forest, AdaBoost, and Rotation Forest. The results show that Firefly Optimization Algorithm (FOA) produced the best results. Using the features selected by FOA, Random Forest outperforms other ML algorithms with 95.36% accuracy and AUC-ROC of 98.9%. This result is followed by Rotation Forest (accuracy 94.89%) and AdaBoost performs the least with accuracy of 91.42%. The findings highlight the efficacy of the FOA meta-heuristic algorithm in phishing website detection.

Keywords: Phishing, security, meta-heuristic, machine learning, ensemble algorithms.

1. Introduction

The increasing adoption of cloud-based and mobile edge technologies has led to the proliferation of web-based and mobile-based solutions that come with essential services to users (Adewole, Akintola, Salihu, Faruk, & Jimoh, 2019; Adewole et al., 2021a; Adewole et al., 2021b; Balogun et al., 2021). There is a continued deployment of Internet-based solutions due to the ease of accessibility and global acceptance of IoT technologies. Of such solution is the deployment of web services and social networking platforms that have emerged in a variety of forms such as Facebook, ChatGPT, Google Bard, WhatsApp, web-based e-commerce applications, e-counselling, e-advert, e-banking, e-hunting websites, and e-health among others (Adil, Khan, & Ghani, 2020). Nevertheless, the continued deployment of these solutions has created an endless possibility for cyber-attackers to explore vulnerabilities and ingress points for malicious intentions. This is possible because of the inability to control the usage of cyberspace (Balogun et al., 2021; Karale, 2021). Therefore, combating cyber-attacks has continuously attracted the attention of both research communities both in academia and industry.

A crucial and severe threat that is associated with web- and mobile-based applications are phishing attacks. This attack

is used to usurp network security and hijack legitimate resources (Adewole et al., 2022; Adil et al., 2020). Today cyberspace has witnessed a rise in illegitimate websites created to collect legitimate users' credentials to compromise the victims' accounts (Vitorino, Andrade, Praça, Sousa, & Maia, 2022; Vrbančič, Fister Jr, & Podgorelec, 2018). Phishing attacks also include fake e-mails, defaced websites, and fake web- and mobile-based solutions which are employed to steal users' credentials or usurp the security of the applications. The website phishing attack is a serious cybersecurity issue with devastating effects on Internet users and web-based businesses (Soon, Chiang, On, Rusli, & Fun, 2020; Wei et al., 2019). Phishing attack has been classified as a serious threat to web-based infrastructures. More importantly, the Anti-Phishing Working Group (APWG) reported the existence of over 51,000 phishing websites in 2018 (Chiew, Tan, Wong, Yong, & Tiong, 2019). In 2021, phishing attacks increased by another 21% in comparison to the 2020 statistics leveraging on global challenges like COVID-19 pandemic, an increase in remote work by employees and the critical Log4j vulnerability (Zscaler, 2022).

To provide countermeasures to phishing attack vectors, different anti-phishing systems have been studied and promised in the literature. The first of its kind is based on

the blacklist method where the crowdsourcing is employed to create labels of suspicious websites as phishing websites. The notable blacklisting database for phishing detection is PhishTank (Cisco Talos Intelligence Group, 2023), URIBL (URIBL, 2023), and Google Safe Browsing (Google, 2023), among others. Blacklist methods compare the universal resource locator (URLs) visited by the user with the stored phishing website URLs to establish a match. This outcome is used to alert the usual of the status of this link. Early phishing detection methods are based on this approach. However, the significant disadvantage of blacklist-based phishing website detection is the failure to identify new phishing links (i.e. lack zero-day phishing attack detection) since blacklist is based on precompiled phishing website URLs. To address this limitation, machine learning (ML) has been in the forefront and the literature is populated with several ML models for phishing website detection (Alazaidah et al., 2024; Taofeek, 2024; van Geest, Cascavilla, Hulstijn, & Zannone, 2024). ML approach provides a dynamic way of addressing phishing website detection problems with a focus on searching for promising ML algorithms or efficient features to aid the detection (Balogun et al., 2021; Yang, Zhao, & Zeng, 2019; Zamir et al., 2020; Zhu, Ju, Chen, Liu, & Fang, 2020). In the effort to identify promising ML algorithms (Zamir et al., 2020), techniques such as tree-based (Adewole et al., 2021b; Balogun et al., 2021), meta/ensemble learning (Balogun et al., 2021; Chiew et al., 2019; Soon et al., 2020), rule-based (Adewole et al., 2019) and streaming-based (Adewole et al., 2021a; Adewole et al., 2022) and deep learning methods (Soon et al., 2020; Vrbanić et al., 2018; Wei et al., 2019; Yang et al., 2019) have been studied. Some feature selection algorithms have also been considered to identify salient attributes for phishing website detection (Adewole et al., 2021b). However, the literature lacks comprehensive studies on the performance of meta-heuristic algorithms for phishing website detection. The dynamism of phishing websites' evasion strategies demands sophisticated analysis of the most discriminating features that are promising for detecting phishing URLs.

Therefore, this paper aims to address this open research issue by providing a comprehensive analysis of ten (10) meta-heuristic algorithms in search of discriminating features for detecting phishing websites. To evaluate the efficacy of the identified features, this paper studied three ensemble-based classifiers: Random Forest, AdaBoost and Rotation forest. More specifically, this paper provides answers to three main research questions:

1. How effective are the selected meta-heuristic algorithms for identifying salient features for phishing website detection?
2. Which ensemble methods among the chosen three algorithms provide promising results when used in combination with the meta-heuristic algorithms?
3. How well is the performance of the proposed approach when compared with the state-of-the-art methods?

The contributions of this paper are manifold. Particularly, this paper:

1. provides an extensive study of ten (10) meta-heuristic algorithms targeted towards phishing website detection;
2. investigates the performance of three ensemble-based classifiers for phishing website detection;
3. provides an empirical comparison of the performance of the proposed approach with the state-of-the-art methods;

The remaining parts of this paper are organized as follows: Section 2 focuses on related works in website phishing detection. Section 3 discusses the proposed methodology adopted in this paper. Section 4 provides a detailed experimental procedure to aid the replication of the results presented in this paper. Section 5 presents the results obtained as well as the analyses of these results. Lastly, Section 6 concludes the paper and offers future research direction in the domain of phishing website detection.

2. Related Works

This section provides a detailed discussion of related works in phishing website detection. As an example, Babagoli, Aghababa, and Solouk (2019) published a meta-heuristic-based nonlinear regression ML phishing website detection model fitted on the University of California, Irvine (UCI) phishing website dataset. The meta-heuristic harmony search technique was implemented to estimate the coefficients of the nonlinear regression model. Phishing websites were detected at an accuracy of 92.8%, based on the harmony search nonlinear regression model. The model was developed using 20 salient features identified through another Decision Tree and wrapper-based models indicating that the meta-heuristic nonlinear regression model is a high-level phishing website detection model liable to higher computational costs.

In another study, (Adewole et al., 2021b), two meta-heuristic (i.e., genetic algorithm and wolf optimization) methods were utilized to enhance the Random Forest (RF) and Support Vector Machine (SVM) phishing website detection models fitted on two distinct phishing website datasets. The RF model enhanced with the wolf optimization meta-heuristic method was reported to accurately detect phishing websites at 94.4%. However, the implemented meta-heuristic methods were unable to increase the accuracy of detecting phishing websites when compared with the 97.2% accuracy of the RF model without feature selection.

The study (Anupam & Kar, 2021) evaluated the performance of four meta-heuristic methods (i.e., Grey Wolf Optimiser (GWO), Bat Algorithm, Firefly Algorithm, and Whale Optimisation Algorithm) used to specifically enhance a SVM classification model for phishing website detection. All the four meta-heuristic methods were considered for finding the optimum parameters for the SVM model. Based on a modified UCI phishing website dataset, where the target variables are grouped into two (i.e., phishing or suspicious, and legitimate) websites and three variations of model training and testing, the SVM model

whose parameters were identified by GWO meta-heuristic was reported to be 90.38% accurate. This was closely followed by the SVM model with BOA meta-heuristic-based parameter with a mean accuracy of 90.21%. The reported accuracy for detecting phishing websites requires improvement, however, the study demonstrates that meta-heuristic methods can also be used for identifying optimal parameters.

Besides implementing classical machine learning classification algorithms to detect phishing websites, an implementation of deep learning was also carried out for the same purpose. In the study (Ali & Ahmed, 2019), a Deep Neural Networks (DNNs) model was hybridized with the Genetic Algorithm (GA) mainly for feature weighting and selection towards the development of a phishing website detection model on the UCI dataset. Based on a 70:30 train-test split model development and evaluation technique, the study reported a DNN accuracy of 88.77%, DNN-GA based feature selection accuracy of 90.39% and DNN-GA based features weighting accuracy of 91.13%. Although the enhancement of the DNN model with the GA meta-heuristic method for detecting phishing websites increased the performance of the DNN model, the highest accuracy reported in the study requires further improvement.

The study in Asiri et al. (2024) proposed PhishingRTDS, a real-time detection system for phishing attacks based on deep learning. Similarly, Asiri et al. (2023a) introduces a deep learning model designed to detect sophisticated phishing attacks, such as Browser-in-the-Browser (BiTB) and malvertising, which often evade traditional detection systems. Unlike conventional methods that focus solely on the main URL, PhishTransformer analyzes all embedded URLs within a webpage, including hyperlinks and iframes. Asari et al. (2023) provides a comprehensive overview of phishing attacks that utilize HTML and URL vectors. It emphasizes the growing sophistication of phishing techniques and the challenges they pose to traditional detection methods. The authors analyze various deep learning-based detection models, focusing on aspects such as data preprocessing, feature extraction, model architecture, and performance metrics.

Overall, various meta-heuristic methods are reportedly used for the identification of salient features towards developing models to accurately detect phishing websites. While some application of meta-heuristics methods (Ali & Ahmed, 2019; Babagoli et al., 2019) was able to enhance the detection of phishing websites by ML models, others (Adewole et al., 2021b) could not significantly improve the accuracy of detecting phishing websites. Importantly, related studies have utilized meta-heuristic to enhance both classical classification and deep learning algorithms for detecting phishing websites. However, a comprehensive study on the performance of meta-heuristic methods to enhance ensemble-based ML phishing website detection models is missing and should therefore be considered.

Khafaga et al. (2023) introduced a novel metaheuristics-based technique for feature selection, which is called binary

Adaptive Squirrel Search Optimization Algorithm (bASSOA). The approach was addressed the instability that may be caused as result of employing metaheuristics for feature selection. Their proposed approach was able to select few number of features while maximizing classification accuracy. To evaluate the performance of this method, authors conducted experiments on a ten-feature dataset from the University of California, Irvine (UCI) repository, comparing it against eleven other state-of-the-art approaches, including binary Grey Wolf Optimization (bGWO), binary Hybrid Grey Wolf and Particle Swarm Optimization (bGWO-PSO), binary PSO (bPSO), binary Stochastic Fractal Search (bSFS), binary Whale Optimization Algorithm (bWOA), binary Modified Grey Wolf Optimization (bMGWO), binary Multiverse Optimization (bMVO), binary Bowerbird Optimization (bSBO), binary Hybrid GWO and Genetic Algorithm (bGWO-GA), binary Firefly Algorithm (bFA), and binary GA methods. The experimental results affirm the superiority and effectiveness of the proposed algorithm in addressing the challenge of feature selection. The experimental outcomes indicate that the suggested bASSOA methodology surpasses alternative feature selection methods, exhibiting an average error rate of 0.63. Moreover, the bASSOA approach demonstrates the most favorable standard deviation fitness of 0.248 in comparison to the other methodologies.

Minocha & Singh (2022) introduced a novel phishing detection system that employs the Binary Modified Equilibrium Optimizer (BMEO) in conjunction with a newly proposed AV-shape transfer function (AV-BMEO) and a k-nearest neighbor classifier. AV-BMEO, characterized by robust exploration and exploitation capabilities, is utilized for both feature selection and optimization of the classifier's hyperparameters. The AV-shape transfer function, developed through opposition-based learning, aims to enhance the algorithm's exploration capabilities. Through statistical validation, the authors reported that AV-BMEO demonstrates superior performance in terms of accuracy and the number of selected features across eighteen datasets when compared to seventeen alternative algorithms.

In enhancing the efficacy of the Genetic Algorithm, Wang (2022) developed an upgrade algorithm called Extended compact genetic algorithm (ECGA). Following fitness calculation, the information entropy of the population is computed to gauge population diversity. To expedite convergence, a consensus mechanism is introduced post-mutation, serving as an alternative mutation operator. Subsequently, ECGA is employed for feature selection to assess its practical impact. Experimental findings demonstrate that the enhanced GA outperforms feature selection algorithms proposed in the last two years for web phishing detection. Their proposed algorithm achieves an average F score of 97%.

Hosseinalipour & Ghanbarzadeh (2022) presented a novel approach to spam detection utilizing the Horse Herd metaheuristic Optimization Algorithm (HOA). Initially, the

continuous HOA is adapted into a discrete algorithm. Subsequently, the inputs of the resulting algorithm are transformed to opposition-based and then converted to multiobjective, making it suitable for addressing the discrete and multiobjective nature of spam detection. Evaluation results demonstrate the superior performance of the proposed method in comparison to other approaches such as K-nearest neighbors-grey wolf optimization, K-nearest neighbors, multilayer perceptron, SVM, and Naive Bayesian. The findings highlight the proposed algorithm's significant accuracy improvement in detecting spam emails within the dataset compared to similar algorithms, coupled with lower computational complexity.

Alsenani et al. (2023) employed Particle Swarm Optimization (PSO) method for feature selection in the identification of phishing websites. Given the critical role of feature engineering in phishing website detection systems and the substantial impact of prior knowledge on detection accuracy, the study utilizes PSO iteratively to enhance the proposed detection model. Various computational methods are incorporated to assess the outcomes. The experimental results revealed significant enhancements in classification accuracy, sensitivity, specificity, f1-score, and Matthew's correlation coefficient through the application of the PSO-based feature selection model. When tested on two prominent datasets comprising both phishing and legitimate websites, the proposed model achieves an accuracy of 97.81% and 90.39% using artificial neural networks in both instances. These findings underscore the importance of judicious feature selection in enhancing the efficiency of phishing detection systems.

Alshammari et al. (2023) employed three classifiers—decision tree, k-nearest neighbors (KNN), and SVM to refine the accuracy of a detection algorithm. The accuracy of detecting a target class is enhanced through the careful selection of pertinent features, determining the class label with the highest probability. This study articulates how feature selection with the Chaotic Dragonfly Algorithm (CDA) yields more precise outcomes compared to all baseline classifiers. It also guides the choice of the most suitable classifier for phishing website detection. The evaluation of the proposed approach involves experimentation across three publicly accessible datasets. The results showed that the KNN and decision tree classifiers demonstrated superior performance on this limited dataset compared to the SVM classifier, achieving high accuracy scores with or without DA.

Jovanovic et al. (2023) proposed a hybrid approach leveraging the eXtreme Gradient Boosting (XGBoost) machine learning model, optimized through an enhanced version of the widely-known metaheuristics algorithm. The study employs an improved firefly algorithm with a two-tier framework, developed as part of this research: to conduct both feature selection and fine-tuning of XGBoost hyperparameters. The performance of this hybrid model was evaluated across three publicly available phishing website datasets, and its efficacy was compared with state-of-the-art metaheuristics operating in the same framework. Two

datasets are sourced from Mendeley Data, while the third is obtained from the University of California, Irvine machine learning repository. Furthermore, the top-performing models undergo SHapley Additive exPlanations (SHAP) analysis to discern the influence of each feature on model decisions. The results indicate that the proposed hybrid solution (97.6029% accuracy) outperforms other approaches.

Bacanin et al. (2023) presented an approach aims to enhance the detection of phishing websites by fine-tuning an extreme learning model that utilizes the most pertinent subset of features from phishing website datasets. To achieve this objective, a novel diversity-oriented social network search algorithm was developed and integrated into a two-tier cooperative framework. This algorithm was compared against six other state-of-the-art metaheuristic algorithms implemented in the same framework and tested under identical experimental conditions. All metaheuristics operated in level 1 of the framework to perform feature selection. The optimal subset of features obtained was then inputted into level 2, where all algorithms fine-tune the extreme learning machine, adjusting parameters such as the number of neurons in hidden layers, as well as weights and biases initialization. For evaluation of this approach, three phishing website datasets of varying sizes and class numbers from UCI and Kaggle repositories were used, with methods compared in terms of classification error for both layers across multiple independent runs. Detailed metrics of the final outcomes, including precision, recall, F1 score, receiver operating characteristics, and precision–recall area under the curves, were also considered. An additional experiment evaluates only layer 2 of the proposed framework to assess metaheuristics' performance in extreme machine learning tuning with all features, representing a large-scale NP-hard global optimization challenge. Statistical tests on the results indicated that the proposed diversity-oriented social network search metaheuristics consistently outperforms competitors for both challenges and all datasets on average. Finally, SHapley Additive exPlanations analysis of the best-performing model is applied to identify the most influential features.

3. Methodology

3.1 Proposed framework

This section presents the research methodology that is adopted in this paper. Figure 1 shows a brief high-level conceptual view of the proposed framework to investigate the performance of the selected meta-heuristic algorithms as well as the ensemble classifiers for detecting phishing websites. As stated in the previous section, the performance of ten (10) meta-heuristics optimization algorithms was investigated. These algorithms include Ant colony optimization algorithm (ACO), Bat optimization algorithm (BOA), Bee colony optimization algorithm (BCO), Cuckoo optimization algorithm (COA), Elephant herding optimization algorithm (EOA), Firefly optimization algorithm (FOA), Flower pollination algorithm (FPA), Genetic algorithm (GA), Rhinoceros optimization algorithm (ROA), and Wolf optimization algorithm (WOA). The

justification for selecting the algorithms is due to their popularity in the literature for addressing optimization problems and guided by their relevance and suitability to the specific demands of phishing detection systems. The Phishing detection is inherently a high-dimensional classification problem characterized by a diverse set of features, including lexical, structural, and behavioural attributes. Therefore, effective feature selection techniques must possess the capability to explore large and complex search spaces to identify compact yet informative subsets of features. The selected algorithms are well-established in addressing such challenges in optimization tasks, particularly in domains with similar complexity.

Furthermore, phishing detection systems must be resilient to evolving attack strategies and adaptive to novel, zero-day threats. The selected meta-heuristic algorithms, many of which are inspired by biological or sociological processes (e.g., swarm intelligence, evolutionary mechanisms), inherently support dynamic search behaviors that balance global exploration and local exploitation. This adaptability is essential for capturing subtle and non-linear feature interactions that often distinguish phishing websites from legitimate ones.

Several of the algorithms—such as GA, FOA, and ACO—have demonstrated efficacy in cybersecurity applications, including spam detection, intrusion detection, and phishing classification. Their proven performance in these domains underscores their potential in enhancing feature selection

and classification accuracy in phishing detection tasks. Notably, many of these algorithms can be adapted to binary or discrete search spaces, which aligns well with the structure of most phishing datasets.

In addition, the algorithms represent a diverse range of search paradigms, encompassing population-based, bio-inspired, and socially influenced strategies. This diversity enables a comprehensive empirical evaluation of different optimization behaviours in the context of phishing detection and mitigates the risk of bias toward a specific class of heuristic methods.

Therefore, this paper seeks to investigate which of these algorithms can produce a subset of features with high-level detection capability for phishing website. To ascertain the performance of each algorithm, three ensemble-based machine learning classifiers, Random Forest, AdaBoost, and Rotation forest were selected. The subsequent sections provide the details of each method.

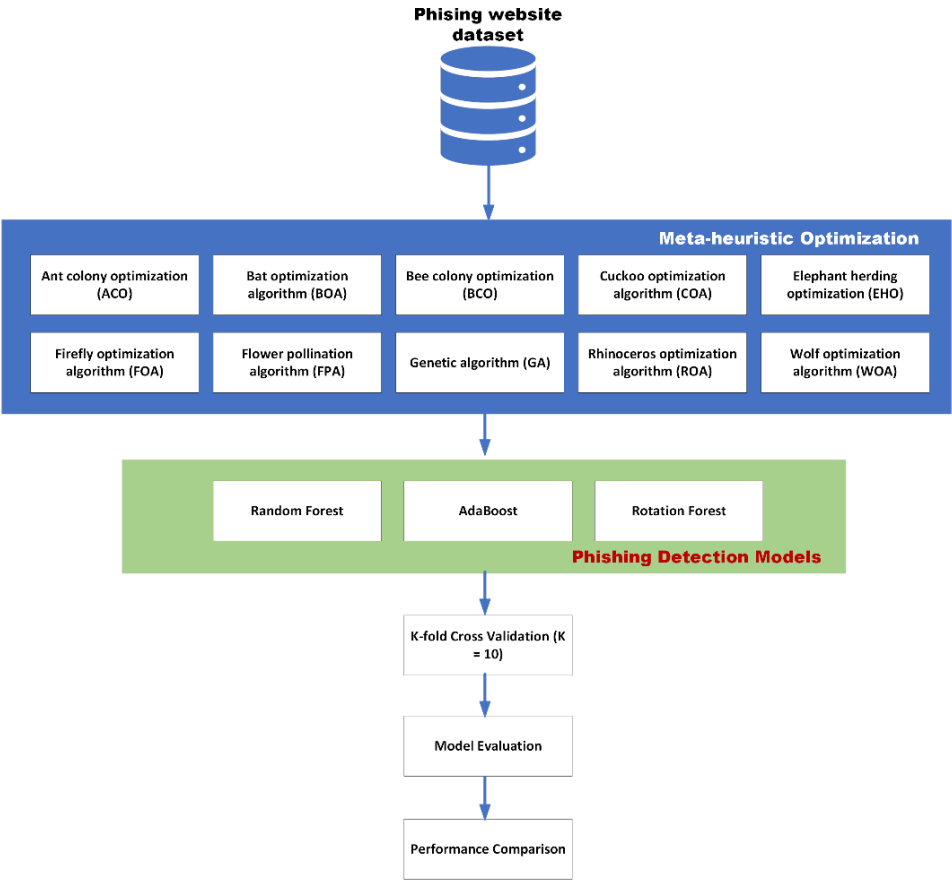


Figure 1: Proposed framework for phishing website detection

3.2 Meta-heuristics optimization algorithms

An important phase in developing an effective machine-learning model is feature selection. Feature selection has been employed in several fields of study including phishing website detection (Adewole et al., 2021b). Feature selection provides an opportunity to construct less complex and comprehensible machine-learning models, improve model performance, and prepare clean and understandable data of good quality. However, the literature lacks a comprehensive evaluation of the performance of meta-heuristic optimization algorithms for phishing website detection.

3.2.1 Ant colony optimization algorithm

Ant Colony Optimization (ACO) is a meta-heuristic algorithm for solving complex combinatorial optimization problems. ACO is based on the basic principle of the laying of pheromone trails, which is inspired by the behaviour of biological ants. Ants use pheromones as a medium of communication. Leveraging this, ACO is based on simple agents such as artificial ants conversing with others indirectly utilizing artificial pheromones within a colony, similar to the biological example. Algorithm 1 gives the basic details of ACO optimization procedures.

Algorithm 1: Ant colony optimization algorithm (ACO)

Input: Instance of the optimization problem

Output: best-solution (S_{best})

PheromoneValuesInitialization(τ)

$S_{best} = \text{NULL}$

while not termination condition **do**

$S_{space} = \emptyset$

for $i = 1$ to n_a **do**

$s = \text{ConstructSoln}(\tau)$

if ValidSoln(s) **then**

$s = \text{LocalSearch}(s)$

if ($f(s) < f(S_{best})$) || ($S_{best} = \text{NULL}$) **then** $S_{best} = s$

$S_{space} = S_{space} \cup \{s\}$

end if

end for

 PheromoneValuesUpdate(τ , S_{space} , S_{best})

end while

return S_{best}

3.2.2 Bat optimization algorithm

Bio-Sonar is a powerful sound propagation method utilized by bats, which are fascinating creatures. Bio-sonar, commonly known as echolocation, is utilized to locate prey and/or navigate over obstacles in the dark. Microbats are among the species that utilize echolocation the most. Bat optimization algorithm (BOA) leverages the advantages of microbats to perform exciting meta-heuristic optimization. The BOA search space can be investigated by adjusting position and velocity vectors (or updated positions vectors). Each bat b_i in a d -dimensional search space can be characterised by its position D_i , frequency F_i , and velocity V_i . Algorithm 2 describes the procedure of BOA.

Algorithm 2: Bat optimization algorithm (BOA)

Input: initialize algorithm parameters

Output: best-solution (S_{best})

Initialize the bat population: D_i ($i = 1, 2, \dots, n$) and V_i

Define pulse frequency F_i

Initialize pulse rate R_i and the loudness L_i

while ($t < Max_{itr}$) **do**

 Generate new solutions by adjusting frequency

 Updating the velocities and positions

if ($\text{rand} > R_i$) **then**

 Select a solution among the best solutions

 randomly

 Generate a local solution around the selected best

 solution

end if

 Generate a new solution by flying randomly

if ($\text{rand} < L_i$ and $f(D_i) < f(S_{best})$)

 Accept the new solutions

 Increase R_i and reduce L_i

end if

 Rank the bats and find the current S_{best}

end while

return S_{best}

3.2.3 Bee colony optimization algorithm

Bee colony optimization (BCO) is a population-based method. A population of artificial bees searches for the best solution. Artificial bees symbolize agents who collectively solve complicated combinatorial optimization problems (Das, Das and Dey, 2018). Each bee generates one solution to the problem. Each bee decides to travel to the source of nectar by following a nestmate who has already located a patch of flowers. Each hive contains a so-called dance floor where bees that have located nectar sources dance to persuade their nestmates to follow them. If a bee decides to leave the hive for nectar, she follows a bee dancer to one of the nectar locations. Upon arrival, the foraging bee collects a load of nectar before returning to the hive and transferring the nectar to a food storer bee. All bees are within the hive at the beginning of the search. Algorithm 3 describes the procedures in detail.

Algorithm 2: Bee colony optimization (BCO)

Input: initialize algorithm parameters

Output: best-solution

Initialize the population: Each bee is set to an empty solution

Randomly Generate n_s scout bees

Evaluate the nectar quantity of each visited location by the scout bee

Sort in descending order the locations visited based on quantity of nectar available

Select the best m visited locations among the n_s visited locations

while (stopping criteria not met) **do**

 Generate n_p neighbourhood search for each selected

```

location
    Evaluate the nectar quantity of each neighborhood
location
    Sort the ( $n_b \times m$ ) sites in descending order according to
    their nectar quantities
    Select the best m visited locations out of ( $n_b \times m$ )
    visited location
end while
return best-solution

```

3.2.4 Cuckoo optimization algorithm

Cuckoo optimization algorithm (COA) is a meta-heuristic based on the compelled brood bloodsucking behaviour of certain cuckoo species as well as the Levy flight actions of certain birds and fruit flies (Mosaad, abed el-Raouf, Al-Ahmar, & Banakher, 2019). The algorithm has been applied in various fields. Algorithm 4 gives the details of COA search.

Algorithm 4: Cuckoo optimization algorithm (COA)

Input: n : initial host nest size, probability $p_a \in [0, 1]$, Max_{itr} : maximum iteration
Output: best-solution

```

Let t = 0 //counter
for i = 1 to n do
    initial population generation for n host  $x_i^{(t)}$ 
    //n is the size of the population
    evaluate fitness function  $f(x_i^{(t)})$ 
end for
while (t <  $Max_{itr}$ ) do
    stochastically generate new Cuckoo solution  $x_i^{(t+1)}$ 
    based on Levy flight
    evaluate fitness function  $f(x_i^{(t+1)})$ 
    stochastically choose nest  $x_j$  from n solutions
    if
         $f(x_i^{(t+1)}) > (f(x_j^{(t)}))$ 
    then
        Replace solution  $x_j$  with  $x_i^{(t+1)}$ 
    end if
    Abandon a fraction ( $p_a$ ) of worst nests
    Create new nests at new location with Levy flight based
    on a fraction  $p_a$  of worst nests
    Keep the nests with best quality (i.e best solutions)
    Let t += 1 //increment counter
end while
return best-solution

```

3.2.5 Elephant herding optimization algorithm

As social beings, elephants inhabit social structures consisting of females and young. A matriarch leads a group of elephants that are collectively known as a clan. The female members prefer to reside with family members, whilst the male members want to reside elsewhere. Gradually gaining independence from their families, they

will ultimately leave them behind. The Elephant Herding Optimization (EHO) algorithm is a recent swarm intelligence algorithm proposed by (Wang, Deb, Gao, & Coelho, 2016). The implementation of EHO algorithm employed simplified elephant herding behaviour. Despite being a relatively new metaheuristic, EHO has been effectively used to solve a variety of problems. Algorithm 5 present a detail description of EHO.

Algorithm 5: Elephant herding optimization (EHO)

Input: n : population size, maximum generation, boundaries
Output: best-solution

```

Set generation counter  $j = 1$ , set maximum generation
 $Max_{Gen}$ 
Initialize the population
Compute elephant's fitness

while (j <  $Max_{Gen}$ ) do
    Sort all the elephants according to their fitness
    for each clans  $c_i$  in the population do
        for each elephants  $h$  in the clan  $c_i$  do
            Update solution(old generation) and generate
            new solution (new generation)
            if new solution = best-solution then
                Update old solution and generate new
                solution
            end if
        end for
    end for
    for each clans  $c_i$  in the population do
        Replace the worst elephant in clan  $c_i$ 
    end for
    evaluate population by the newly updated positions
    increment j
end while
return best-solution

```

3.2.6 Firefly optimization algorithm

Yang (2009) proposed the idea of firefly optimization. This algorithm relies on the facts that firefly is a unisex specie which make all the fireflies attracted to one another irrespective of their sex. This attractiveness is proportional to the brightness so any interaction between two fireflies, the less bright firefly will move to the brighter one. This attractiveness reduces based on the distance between them. Random movement is employed when there is no brighter firefly than the other. The brightness is influenced by the fitness function and can be proportional to the fitness function. Algorithm 6 further describes the operations of firefly algorithm.

Algorithm 6: Firefly optimization algorithm (FOA)

Input: n : population size – the number of fireflies, Max_{itr} : maximum iteration

γ : parameter for attractiveness, α : random term contribution

Output: best-solution (S_{best})

```

initial population generation for n fireflies randomly
positioned
obtain  $S_{best}$  based on fitness
while (not stopping criteria) do
  for each firefly i in fireflies do
    for each firefly j in fireflies do
      if brighter(firefly j, firefly i) then
        position firefly i towards j with fitness
function
      end if
    end for
  end for
  compute  $S_{best}$  by evaluating the position of individual
fireflies
end while
return  $S_{best}$ 

```

3.2.7 Flower pollination algorithm

Flower pollination algorithm (FPA) is based on the natural phenomena of flower pollination, which considered both biotic and abiotic forms of pollination. The biotic and cross-pollination is considered the global pollination procedure that are responsible for Levy flights. The abiotic and self-pollination are local pollination while the flower constancy is taken as the reproduction probability that is proportional to the similarity of two flowers in the pollination process. A probability called switch probability is used to adjust the local and global pollination. This algorithm is further described in Algorithm 7.

Algorithm 7: Flower pollination algorithm (FPA)

Objective: min or max

$f(x), x = (x_1, x_2, \dots, x_d)$

Input: n : population size – the number of flowers with random solutions, Max_{itr} : maximum generation

p : switch probability $\in [0, 1]$

Output: best-solution (S_{best})

Search for the (S_{best}) in the initial population generated

```

while ( $t < Max_{itr}$ ) do
  for  $i = 1$  to  $n$  do
    if  $rand < p$  then
      Randomly obtain d-dimensional vector that obeys
      the Levy distribution
      Compute Global pollination
    else
      Randomly obtain  $\epsilon$  based on a uniform distribution
       $[0, 1]$ 
      Select j and k randomly from all the solutions
      Compute Local pollination
    end if
    Evaluate new solutions
    If new solutions give better results, update them in
    the population
  end for
  Search for the current ( $S_{best}$ )
end while
return  $S_{best}$ 

```

3.2.8 Genetic algorithm

The genetic algorithm (GA) is a popular natural selection-inspired optimization algorithm. It is a population-based search algorithm that employs the principle of survival of the fittest. The fresh populations are generated by repeatedly applying genetic operators to existing individuals. The key components of GA as presented in Algorithm 8 are the chromosome depiction, crossover, selection, fitness function, and mutation computation.

Algorithm 8: Genetic algorithm (GA)

Input: n : population size, Max_{itr} : maximum iteration

Output: best-solution (S_{best})

initial population generation of n chromosomes

$Y_i \forall i = 1, \dots, n$

Let $t = 0$ //counter

Calculate fitness value for each chromosomes

while ($t < Max_{itr}$) **do**

select a pair of chromosomes from initial population based on fitness

apply crossover operation on the chosen pair based on crossover probability

apply mutation operation on the offspring based on mutation probability

replace old population with newly generated population

Let $t += 1$ //increment counter

end while

return S_{best}

3.2.9 Rhinoceros optimization algorithm

The natural behavior of rhinoceros has been leveraged to develop robust meta-heuristic optimization algorithm called Rhinoceros optimization algorithm (ROA). The basic idea explores distribution difference among the gender distinct roles of rhinoceros which gives opportunity to model inclination of global and local explorations. ROA made certain assumptions about rhinoceros's behavior that include the facts that all agents are performing levy flight within a pre-defined search range; male agents possess bounce strategy and each iteration of the ROA a probability 0.05 indicating possibility that every agent will die or reborn is incorporated. To achieve better computation speed, the group leader is usually abandoned in the assumptions. Algorithm 9 presents a detailed description of ROA.

Algorithm 9: Rhinoceros optimization algorithm (ROA)

Input: mR - population size for male rhinoceros, fR - population size for female rhinoceros

md - male rhinoceros search range radius; fd – equivalent female radius

Max_{itr} : maximum iteration

Output: best-solution (S_{best})

Initialize all rhinocross and parameters

```

while ( $t < Max_{itr}$  and not global best solution obtained) do
  for each male rhinoceros  $mr$  in  $mR$  do
    if  $dist(mr_i, mr_j) < md$  //crown avoidance of male rhinocross
      weak_rhinoceros_escape( $mr.weakest$ );
    end if
    // compute new location
     $mr_i = Levy\ Flight(mr_{t-1})$ 
    update fitness for the male rhinoceros (i.e  $mr$ );
  end for
  for each female rhinoceros  $fr$  in  $fR$  do
    // compute new location
     $fr_i = Levy\ Flight(fr_{t-1})$ 
    update fitness for the female rhinoceros (i.e  $fr$ );
  end for
  forall rhinocross ( $mr$  in  $mR$ ) and ( $fr$  in  $fR$ ) do
    if  $life\_test(r_i) = 0$  then
      remove_rhinoceros( $r_i$ )
      baby_rhinoceros_born( $r_i$ )
    end if
    //use all rhinoceros to find the global_best
    if  $fitness(S_{best}) < fitness(global\_best)$  then
       $global\_best = S_{best}$ 
    end if
    increase counter for  $t$ 
  end while
return  $S_{best}$ 

```

3.2.10 Wolf optimization algorithm

The Wolf Optimization Algorithm (WOA) is a meta-heuristics algorithms based on wolf preying behavior. The algorithm possesses local search capability and as well as flocking movement. Each wolf in WOA explores autonomously by memorizing its attribute and unites with its peer when the peer is in a better location. Based on this behavior, long-range inter-communication amongst the wolves that characterize the searching points for candidate solutions is removed since wolves stalk their target in stillness. WOA delegates its swarm behavior to each individual wolf rather than to a single leader. WOA operates as if there are multiple leaders swarming from different directions to the best solution, instead of a one-directional for optimum solution by a single flock. The algorithm is described in Algorithm 10.

Algorithm 10: Wolf optimization algorithm (WOA)

Input:

objective function:

$f(x), x =$

$(x_1, x_2, \dots, x_d)^T$

wolves population initialization, $x_i \forall i = 1, \dots, w$

Output: best-solution

initialize radius of visual range (r)

initialize step size for wolf movement (s)

initialize wolf velocity factor (α)

define user-defined threshold $p_a \in [0, 1]$ for enemy appearance

```

while ( $t < \text{generations and stopping criteria not met}$ ) do
  for  $i = 1$  to  $w$ 
    preynewfood_initiatively()
    generatenewlocation()
    //check for quality of the next location generated by
    //random number negator. If not new location, repeat
    random location generation
    if
       $dist(x_i, x_j) <$ 
      ( $r$  and  $x_j$ 
      is better such that  $f(x_i) < f(x_j)$ ) then
         $x_i$  moves towards  $x_j$ 
      else
         $x_i = \text{preynewfood\_passively}()$ 
      end if
      generatenewlocation();
      if ( $rand() > p_a$ ) then
         $x_i = x_i +$ 
         $rand() + v$ 
        //new position escape
      end if
    end for
  end while
return best-solution

```

3.3 Ensemble-based classifiers

Ensemble-based classifier is a supervised learning approach introduced by Opitz & Maclin, (1999) which employs multiple learning models to enhance predictive performance. Its overall effectiveness typically surpasses that of individual models, resulting in superior performance compared to standalone learning methods. Among various ensemble learning techniques, Forest is widely recognized, combining weak learners (trees) to create a more accurate and robust learner. Based on the sampling method, strong learners can be broadly categorized into two types: bagging (Breiman, 1996) and boosting (Freund & Schapire, 1996).

The fundamental concept behind bagging involves randomly sampling and then collectively voting on the training sample set, with Random Forest (Breiman, 2001; Hambali et al., 2022) being a prominent example. On the other hand, boosting focuses on adjusting weights in the sampling process, represented by algorithms such as

gradient boosting, gradient-boosted decision tree (GBDT) (Friedman, 2001), XGBoost (Chen & Guestrin, 2016), among others. These algorithms have achieved numerous accolades in recent Kaggle contests and demonstrated excellent performance across various application domains (Turska et al., 2021). However, it is worth noting that evaluating the performance of ensemble learning models typically entails higher computational costs compared to individual learning models.

3.3.1 Random forest

A widely recognized method for classification in various domains is the Random Forest (RF) (Geetha et al., 2019; M. A. Hambali et al., 2022; Seera & Lim, 2014). RF, also known as bagged decision trees (Biau, 2012), involves a collection of weak classifiers to create a robust learner (Geetha et al., 2019). The RF method randomly selects a subset to build an independent decision tree, and the process iteratively splits the randomly chosen subset from the root node to a child node (Suresh et al., 2020). The splitting continues until each tree reaches a leaf node without pruning. Each individual tree models the classification of samples, and the objective variable significantly contributes to the final tree class (vote) (Suresh et al., 2020). The overall classification is determined by majority voting based on the results from all the trees.

3.3.2 AdaBoost

Consequently, numerous researchers have successfully applied the AdaBoost algorithm to address classification problems, including but not limited to object detection tasks such as face recognition, video sequences, and signal processing systems (Hambali et al., 2019). The AdaBoost algorithm commences by assigning weights to the training set, denoted as $(u_1, v_1), \dots, (u_n, v_n)$, where each u_i belongs to the instance space U , and each label v_i is in the label set V , equivalent to $\{-1, +1\}$. Initially, it sets the weight of each training example i in round t as $D_{t(i)}$, with the starting point being $(D_{t(i)} = 1/N, i = 1, \dots, N)$. Subsequently, the algorithm increases the weight of misclassified examples from the base learning algorithm (referred to as a weak hypothesis) to focus on challenging examples within the training set in each round. Algorithm 11 resents the procedures used in AdaBoost algorithm.

Algorithm 11: Adaboost

Input: TS : training set, $TS = u_i (i = 1, 2, \dots, n)$, labels $v_i \in V$
 t : Iteration number

Steps:

1. Assign TS sample $(u_1, v_1), \dots, (u_n, v_n); u_i \in U, v_i \in \{-1, +1\}$
 2. Initialise the weights of $D_1(i) = 1/N, i = 1, \dots, N$
 3. for $t = 1, \dots, T$
 4. Train weak learner using distribution D_t
 5. Get weak hypothesis $h_t: U \rightarrow \{-1, +1\}$ with its error: $\epsilon_t = \sum_{i=h_t(u) \neq v_i} D_t(i)$
 6. Update distribution $D_t: D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i T_t(u_t))}{c_t}$
 7. Next t that, $t+1$
 8. Output the final hypothesis: $H(u) = \text{sign}$
-

3.3.3 Rotation Forest

Rotation Forest (Rodriguez et al., 2006) is an ensemble learning technique that relies on principal component analysis (PCA) for feature extraction. The rotation operation is employed to maintain classifier performance while generating diverse training subsets, ultimately leading to improved diversity. Consequently, employing the same voting process as Forest, Rotation Forest demonstrates superior prediction performance (Lee et al., 2022). Beyond its application in tabular data classification, Rotation Forest has proven effective in image classification tasks (Colkesen & Kavzoglu, 2016; He et al., 2019).

Algorithm 12: Rotation Forest

Input: Training dataset D
 Number of base classifiers K
 Number of features F
 Number of principal components M

Output: Ensemble of K rotated decision trees

Steps:

1. Split the training dataset D into K subsets $D_1, D_2, \dots, D_K$.
2. **For** each subset D_i , perform the following steps:
 - a. Apply PCA to reduce the dimensionality of the features to M principal components.
 - b. Train a decision tree classifier T_i on the transformed subset D_i .
3. Rotate each decision tree T_i by applying a rotation operation to its principal components.
 - a. Randomly select a rotation matrix R_i for each decision tree T_i .
 - b. Rotate the principal components of T_i using the selected rotation matrix R_i .
4. Combine the rotated decision trees to form the ensemble of classifiers.
5. Classification:
 - a. **For** a new instance x :
 - i. Transform x using PCA to obtain its principal components.
 - ii. Classify x using each rotated decision tree in the ensemble.
 - iii. Aggregate the individual tree predictions, e.g., through voting, to make the final prediction.

Output the ensemble of K rotated decision trees.

End Algorithm

In this study, Random Forest, AdaBoost, and Rotation Forest were selected as the ensemble classifiers for evaluating the effectiveness of meta-heuristic feature selection algorithms in phishing website detection. These classifiers were purposefully chosen to represent the three core paradigms of ensemble learning—bagging, boosting, and feature-space transformation. Random Forest, a widely adopted bagging technique, is known for its robustness,

ability to handle high-dimensional data, and strong generalization performance, making it well-suited for classification tasks in cybersecurity domains. AdaBoost, a boosting algorithm, sequentially improves weak learners by focusing on previously misclassified instances, offering a complementary perspective that emphasizes model bias reduction. Rotation Forest, on the other hand, enhances diversity among base classifiers, often yielding superior predictive accuracy in complex classification scenarios. The inclusion of these three ensemble methods allows for a comprehensive evaluation of how different ensemble strategies respond to optimized feature subsets derived from meta-heuristic algorithms. Moreover, their selection is supported by their proven success in prior phishing detection studies and their availability in widely used machine learning platforms such as WEKA and scikit-learn, thereby ensuring reproducibility and practical relevance. Collectively, these classifiers provide a balanced and representative benchmark for assessing the impact of meta-heuristic-driven feature selection on phishing website detection performance.

4. Experimental setup

This section provides a detailed discussion on the experimental settings to detect phishing websites. It includes a discussion on the dataset used, hyperparameters of the meta-heuristic algorithms and the evaluation metrics employed to assess the performance of the various methods. As stated earlier, ten (10) meta-heuristic optimization algorithms were benchmarked to investigate their usefulness in generating minimal features that can provide comparable performance to the state-of-the-arts. Three algorithms have been evaluated with a focus on developing promising ensemble classification model for phishing website detection.

The various experiments in this study have been conducted using WEKA machine learning tool. This tool is widely used in different domains for experimental simulation of ML algorithms. The experiments were performed on a Dell Laptop with Windows 11 operating system, having a random-access memory (RAM) of 32 GB and 2.90 GHz Intel Core i9 CPU with 1 TB hard disk. To train the selected classification algorithms, cross-validation method based on 10-fold was adopted on the phishing dataset described in the subsequent section. All the ML algorithms have been trained using the default hyperparameter settings according to the WEKA tool.

4.1 Dataset description

In this study, the phishing website dataset available on the UCI public repository was employed. This dataset has a total of 31 features including the class attribute and 11,055 instances corresponding to the individual URL records. This dataset is available for download at “<https://archive.ics.uci.edu/dataset/327/phishing+websites>”. Adewole et al. (2019) gives a detailed description of the different features available in this dataset.

4.2 Hyperparameters for meta-heuristic algorithms

To achieve comparable performance across the ten selected meta-heuristic algorithms, different hyperparameters were defined as highlighted in Table 1. These hyperparameters provide objective evaluation of the algorithms taking into account their various requirements. GA has the least configured hyperparameters among the selected algorithms for evaluation.

Table 1: Hyperparameters of the meta-heuristic algorithms

Meta-heuristic algorithms	Hyperparameter
ACO	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Initial tau: 0.1, Evaporation rate: 0.9, Pheromone rate: 2.0, Heuristic rate: 0.7, Report frequency: 20, Seed: 1
BOA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Report frequency: 20, Frequency: 0.5, Loudness: 0.5, Seed: 1
BCO	Start set: 10, Population size: 30, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Radius Damp: 0.98, Radius Mutation: 0.8, Report frequency: 20, Seed: 1
COA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Pa Rate: 0.25, Sigma Rate: 0.69657, Mutation probability: 0.01,

	Report frequency: 20, Report frequency: 20, Seed: 1
EOA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Report frequency: 20, Seed: 1
FOA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Beta zero: 0.33, Absorption coefficient: 0.001, Report frequency: 20, Seed: 1
FPA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Pollination Rate: 0.33, Mutation probability: 0.01, Report frequency: 20, Report frequency: 20, Seed: 1
GA	Start set: 10, Population size: 20, Number of generations: 20, Probability of crossover: 0.6, Probability of mutation: 0.033, Report frequency: 20, Random number seed: 1
ROA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits, Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Report frequency: 20, Seed: 1
WOA	Start set: 10, Population size: 20, Number of iterations: 20, Mutation type: bit-flip, Accelerate type: normal, Objective type: merits,

	Chaotic population type: normal, Chaotic parameter type: normal, Logistic map coefficient: 4.0, Mutation probability: 0.01, Beta zero: 0.33, Absorption coefficient: 0.001, Report frequency: 20, Seed: 1
--	---

4.3 Evaluation metrics

The evaluation metrics utilized for assessing the performance of the classification algorithms include:

Accuracy: Accuracy is a crucial metric for evaluating machine classification models, representing the proportion of correctly predicted instances out of the total instances in the dataset. In the context of binary classification, accuracy can be computed for positive and negative instances.

Precision: Precision measures the accuracy of the classifier and is the percentage of correctly predicted positive instances relative to the total predicted positive instances.

Sensitivity or Recall: Sensitivity or Recall signifies the correctness of predictions for instances belonging to the positive class.

F-Measure (or F-score): F-Measure represents the harmonic mean of precision and recall, combining these metrics into a single score.

AUC-ROC score: Area Under Curve - Receiver Operating Characteristic Curve (AUC-ROC) score is a singular metric that encapsulates the classifier's effectiveness across various classification thresholds. This score is obtained by evaluating the area under the ROC curve. The AUC-ROC score serves as an indicator of the classifier's ability to differentiate between positive and negative classes, and it ranges from 0 to 1.

Matthew's correlation coefficient (MCC) is a statistical measure for assessing models. MCC serves as a tool to quantify the disparity between predicted values and actual values, analogous to chi-square statistics for a 2 x 2 contingency table.

False positive rate (FPR) serves as an accuracy metric for various tests, in machine learning models. Technically, the false positive rate is characterized as the probability of incorrectly rejecting the null hypothesis.

5. Results and discussions

5.1 Feature selection results

Table 2 showcases the features selected by different meta-heuristic algorithms for phishing website detection. It provides a starting point for understanding how different meta-heuristic algorithms approach feature selection for phishing website detection. It highlights the importance of common features like URL characteristics, security aspects, and website traffic, while also showcasing that some algorithms might identify additional informative features. The number of features chosen varies significantly across algorithms. BOA, FOA, and GA select the most features (12), while BCO and FPA choose the least (6). This could be due to the inherent nature of each algorithm's search strategy. Several features appear frequently across multiple algorithms, suggesting their potential importance in

identifying phishing websites. These include URL Features (URL length, presence of subdomain, presence of anchor tags in URLs), Security Features (SSL certificate status), Traffic Features (Website traffic), and Other Features such as Pop-up windows and presence of links pointing to the page.

Table 2: Results of feature selection using Meta-heuristics algorithms

Meta-heuristics algorithms	Number of selected features	Selected features
ACO	8	URLURL_Length, having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, Redirect, web_traffic, Links_pointing_to_page
BOA	12	URLURL_Length, having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, Redirect, on_mouseover, RightClick, popUpWidnow, web_traffic, Page_Rank, Links_pointing_to_page
BCO	6	URLURL_Length, having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, web_traffic
COA	7	URLURL_Length, having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, popUpWidnow, web_traffic
EOA	7	URLURL_Length, having_Sub_Domain, SSLfinal_State, port, URL_of_Anchor, Links_in_tags, web_traffic
FOA	12	URLURL_Length, having_Sub_Domain, SSLfinal_State, Domain_registration_length, URL_of_Anchor, Links_in_tags, Submitting_to_email, Redirect, popUpWidnow, web_traffic, Page_Rank, Links_pointing_to_page
FPA	6	URLURL_Length,

		having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, web_traffic
GA	12	URLURL_Length, having_At_Symbol, Prefix_Suffix, having_Sub_Domain, SSLfinal_State, Favicon, URL_of_Anchor, Links_in_tags, Submitting_to_email, RightClick, popUpWidnow, web_traffic
ROA	8	URLURL_Length, having_Sub_Domain, SSLfinal_State, URL_of_Anchor, Links_in_tags, age_of_domain, web_traffic, Page_Rank
WOA	9	URLURL_Length, having_Sub_Domain, SSLfinal_State, port, URL_of_Anchor, Links_in_tags, popUpWidnow, Iframe, web_traffic

The algorithm-specific feature selection of **BOA** and **FOA** select a higher number of features (12 each) and include additional features like Page Rank and Domain registration length, which might be informative for identifying phishing attempts. **GA** chooses features like Prefix/Suffix in URLs and presence of Favicon, which could be relevant but not as widely selected by others. **EOA** uniquely selects the port feature, potentially indicating its importance for specific phishing tactics.

5.2 Results based on Random Forest

The performance of Random Forest as the ensemble classifier used with various meta-heuristic algorithms for feature selection in phishing website detection is summarized in Table 3. Random Forest achieves good overall performance across all metrics with most accuracy scores exceeding 95%. This suggests that it is a strong candidate for phishing website detection.

Table 3: Results of Random Forest with Meta-heuristic Algorithms

Algorithm/Metric	Accuracy	Precision	Recall	F-Measure	MCC	AUC-ROC	FPR
ACO	94.3736	0.944	0.944	0.944	0.886	0.985	0.060
BOA	94.7806	0.948	0.948	0.948	0.894	0.988	0.054
BCO	93.4057	0.934	0.934	0.934	0.866	0.983	0.071
COA	93.5414	0.936	0.935	0.935	0.869	0.984	0.069
EOA	93.5776	0.936	0.936	0.936	0.870	0.984	0.069
FOA	95.3596	0.954	0.954	0.954	0.906	0.989	0.048
FPA	93.4057	0.934	0.934	0.934	0.866	0.983	0.071
GA	94.4821	0.945	0.945	0.945	0.888	0.988	0.058
ROA	94.1474	0.941	0.941	0.941	0.881	0.986	0.062
WOA	93.7133	0.937	0.937	0.937	0.873	0.984	0.068

The choice of the meta-heuristic algorithm for feature selection can significantly influence the performance of Random Forest. The combination of **Firefly Optimization Algorithm (FOA)** for feature selection and Random Forest for classification achieves the highest accuracy (95.36%), precision (0.954), recall (0.954), F-Measure (0.954), and AUC-ROC (0.989) scores. It also has the lowest False Positive Rate (FPR) of 0.048, indicating it minimizes misclassifying legitimate websites as phishing. Other algorithms such as BOA comes in second with promising performance across most metrics. ACO, GA, COA, and EOA achieve competitive results as well. While BCO, FPA, and ROA have slightly lower performance compared to others.

Feature selection plays a crucial role in improving the performance of Random Forest for phishing website detection. FOA seems to be the most effective algorithm in selecting informative features for this task. While FOA leads to the best results, other algorithms like BOA, ACO, and GA might offer a good balance between performance and simplicity depending on the specific application requirements.

5.3 Results based on AdaBoost

The performance of AdaBoost as the ensemble classifier used with various meta-heuristic algorithms for feature

selection in phishing website detection is shown in Table 4. Surprisingly, unlike Random Forest, the results for

AdaBoost are remarkably consistent across all metrics and feature selection algorithms. Every single entry has the same values for accuracy (91.42%), precision (0.915), recall (0.914), F-Measure (0.914), MCC (0.826), and AUC-ROC (0.975) except for GA. This indicates that the choice of feature selection algorithm has minimal impact on the performance of AdaBoost in this specific case. While AdaBoost still shows a consistent performance across most metrics with other feature selection algorithms (accuracy of 91.42%), GA stands out with a slightly higher accuracy of 91.98%. This suggests that GA might introduce a small improvement in correctly classifying websites compared to other algorithms used with AdaBoost. The feature set chosen by GA for AdaBoost is slightly more informative for differentiating phishing websites from legitimate ones and It is possible that the features selected by GA align better with how AdaBoost weights features during its learning process, leading to a slight improvement in accuracy.

Compared to Random Forest (as discussed in Table 3), AdaBoost shows significantly lower performance across all metrics. This suggests that AdaBoost might not be the best choice for phishing website detection in this scenario. While feature selection algorithms can significantly improve the performance of Random Forest for phishing website detection, they seem to have minimal impact on AdaBoost in this case and AdaBoost achieves a lower overall performance compared to Random Forest for this task.

Table 4: Results of AdaBoost with Meta-heuristic Algorithms

Algorithm/Metric	Accuracy	Precision	Recall	F-Measure	MCC	AUC-ROC	FPR
ACO	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
BOA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
BCO	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
COA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
EOA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
FOA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
FPA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
GA	91.9765	0.920	0.920	0.920	0.837	0.978	0.087
ROA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093
WOA	91.4247	0.915	0.914	0.914	0.826	0.975	0.093

5.4 Results based on Rotation Forest

Rotation Forest achieves good overall performance across most metrics, with accuracy scores exceeding 94% for most entries as shown in Table 5. However, it falls slightly short of the best results achieved by Random Forest. Similar to Random Forest, the choice of the meta-heuristic algorithm for feature selection can influence the performance of Rotation Forest. The combination of Firefly Optimization Algorithm (FOA) for feature selection and Rotation Forest

for classification achieves the highest accuracy (94.89%), precision (0.949), recall (0.949), F-Measure (0.949), and AUC-ROC (0.987) scores. It also has a low False Positive Rate (FPR) of 0.054. Other Algorithms such as BOA comes in second with impressive performance on most metrics. ACO, GA, and some other algorithms achieve competitive results as well. While BCO, FPA, ROA, and WOA have slightly lower performance compared to others.

Table 5: Results of Rotation Forest with Meta-heuristic Algorithms

Algorithm/Metric	Accuracy	Precision	Recall	F-Measure	MCC	AUC-ROC	FPR
ACO	94.3736	0.944	0.944	0.944	0.886	0.984	0.060
BOA	94.4279	0.944	0.944	0.944	0.887	0.986	0.059
BCO	93.3152	0.933	0.933	0.933	0.864	0.981	0.072
COA	93.27	0.933	0.933	0.933	0.864	0.982	0.073
EOA	93.4057	0.934	0.934	0.934	0.866	0.982	0.072
FOA	94.8892	0.949	0.949	0.949	0.896	0.987	0.054
FPA	93.3152	0.933	0.933	0.933	0.864	0.981	0.072
GA	94.0751	0.941	0.941	0.941	0.880	0.985	0.063
ROA	93.8308	0.938	0.938	0.938	0.875	0.984	0.066
WOA	93.4962	0.935	0.935	0.935	0.868	0.983	0.070

Feature selection plays a crucial role in improving the performance of Rotation Forest for phishing website detection, similar to Random Forest. Again, FOA seems to be the most effective algorithm in selecting informative features for this task. While FOA leads to the best results, other algorithms like BOA, ACO, and GA might offer a good balance between performance and simplicity depending on the specific application requirements.

Compared to Random Forest, Rotation Forest shows slightly lower performance overall, suggesting Random Forest might be a more suitable choice for this particular dataset.

5.5 Results comparison for all meta-heuristics and classification algorithms

5.5.1 Based on Accuracy

Figure 2 shows the results of comparing ten meta-heuristic algorithms and three ensemble classification algorithms for phishing website detection. The performance is measured by accuracy. The findings based on the Figure 2 reveals that Random Forest achieved the highest accuracy (95.36%) followed by Rotation Forest (94.89%). AdaBoost had the lowest accuracy (91.42%) among the three ensemble algorithms.

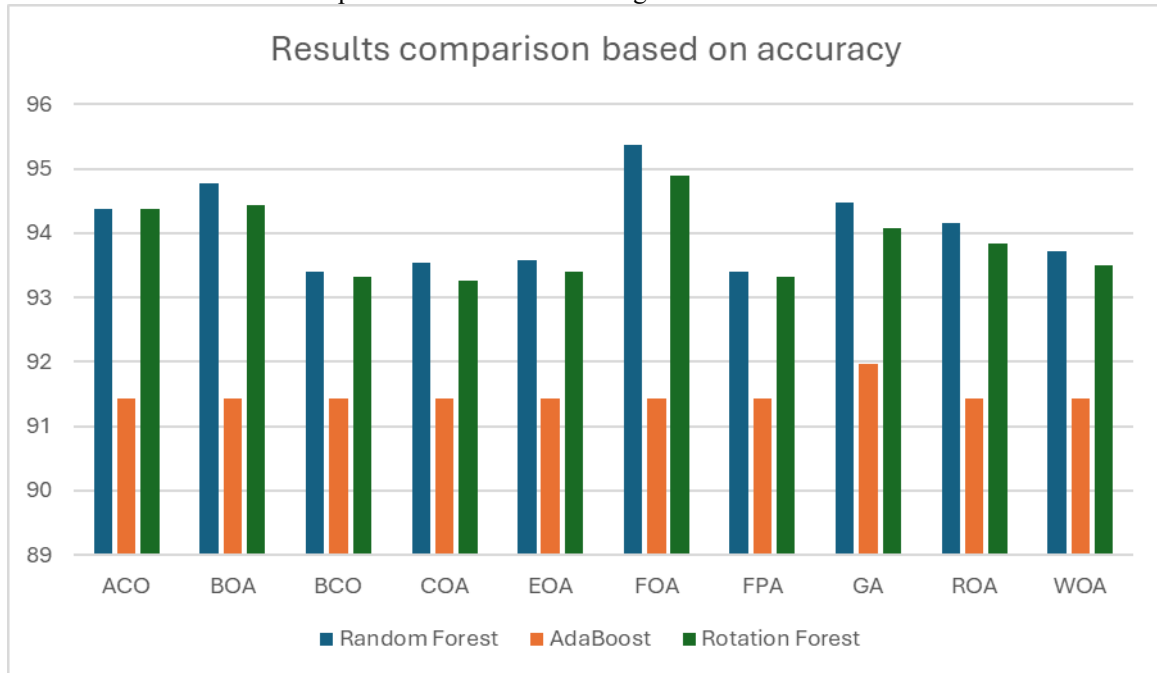


Figure 2: Results comparison based on accuracy.

Meta-heuristic Algorithms for Feature Selection such as FOA (Firefly Optimization Algorithm) produced the best results (95.36% accuracy with Random Forest). BOA (Bat Optimization Algorithm) and GA (Genetic Algorithm) come in second and third place, respectively (both with Random Forest). Overall, the combination of FOA for feature selection and Random Forest for classification achieved the highest accuracy in phishing website detection based on this dataset.

5.5.2 Based on F-Measure

The figure 3 summarizes the F-Measure scores for different combinations of ensemble classifiers (Random Forest, AdaBoost, Rotation Forest) and meta-heuristic feature selection algorithms (ACO, BOA, BCO, etc.).

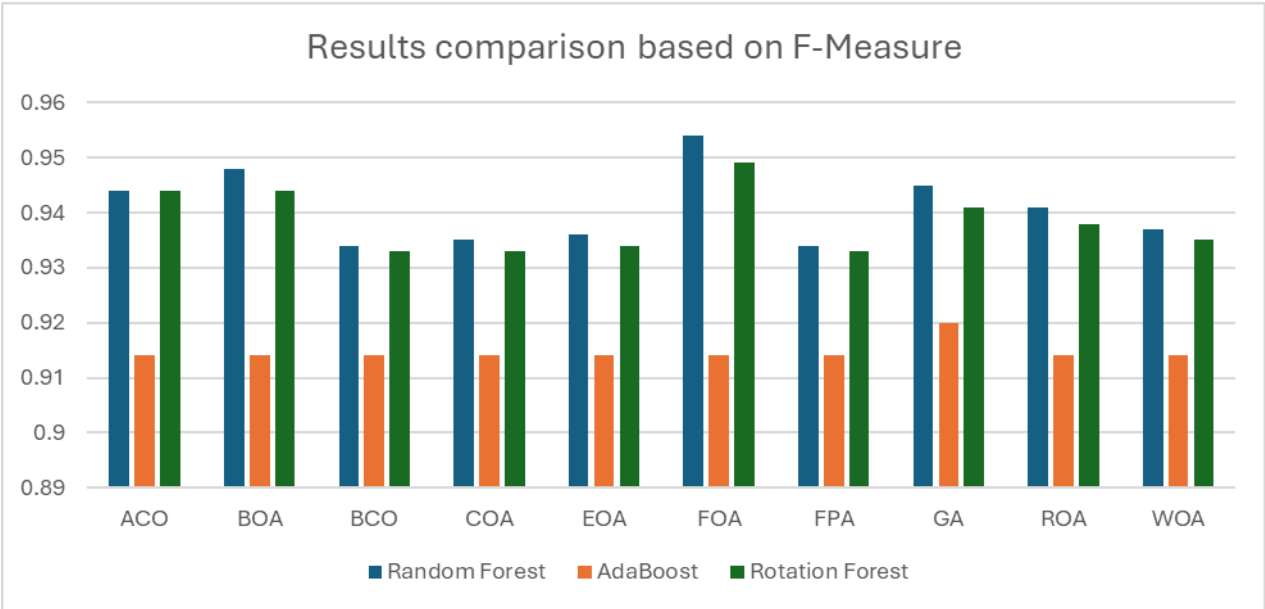


Figure 3: Results comparison based on F1-Measure

The combination of Firefly Optimization Algorithm (FOA) for feature selection and Random Forest for classification achieved the highest F-Measure score of **0.954**, this was identified to be the top performer. Interestingly, comparison with Random Forest, FOA with Random Forest outperforms Random Forest on its own (which had the highest accuracy based on the previous results shows in Figure 2). This suggests that FOA effectively selects features that improve the balance between precision and recall in Random Forest for phishing website detection.

In addition, based on Ensemble Classifier Performance, Random Forest performs best among the three ensemble classifiers with most F-Measure scores above 0.94. AdaBoost consistently has the lowest F-Measure across all feature selection algorithms. Finally, findings based on meta-heuristic algorithm performance shows that FOA stands out as the best performing meta-heuristic algorithm for feature selection based on F-Measure scores. Other algorithms like BOA and GA achieve competitive scores.

These findings suggest that combining FOA for feature selection with a Random Forest ensemble classifier is a promising approach for phishing website detection based on F1-Measure. It achieves a good balance between precision and recall, potentially leading to a more robust detection system.

5.5.3 Based on AUC-ROC

Figure 4 shows the AUC-ROC scores for different combinations of ensemble classifiers (Random Forest, AdaBoost, Rotation Forest) and meta-heuristic feature selection algorithms (ACO, BOA, BCO, etc.). AUC-ROC measures the overall performance of a classification model at all classification thresholds. The findings are summarized as follows:

- Prime Performers: The amalgamation of FOA for feature selection along with Random Forest for classification exhibits the highest AUC-ROC score of 0.989, mirroring the outcomes observed in F-Measure results.
- Ensemble Classifier Performance: Again, Random Forest showcases superior performance among the three ensemble classifiers, boasting AUC-ROC scores predominantly exceeding 0.98. Conversely, AdaBoost consistently registers the lowest AUC-ROC scores across all feature selection algorithms.
- Meta-heuristic Algorithm Performance: In line with F-Measure findings, FOA emerges as the most robust meta-heuristic algorithm for feature selection, as evidenced by AUC-ROC scores. Nonetheless, the overall disparity among algorithms is narrower compared to F-Measure

outcomes. Notably, most of the algorithms achieve commendable AUC-ROC scores surpassing 0.98.

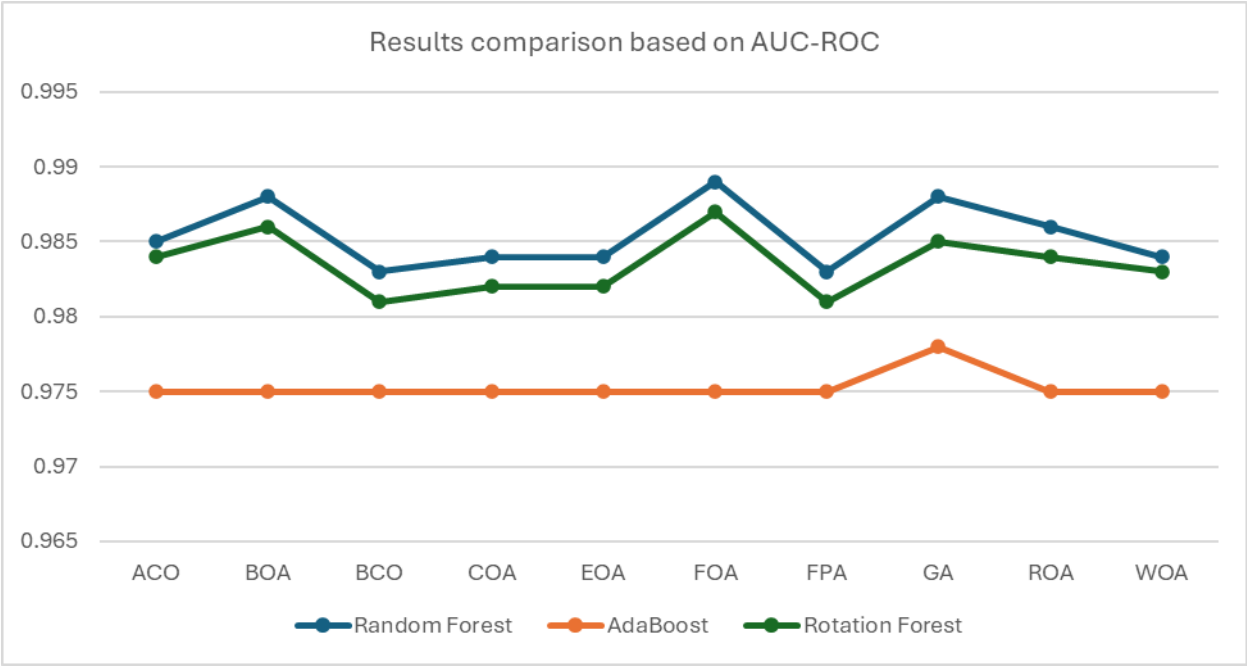


Figure 4: Results comparison based on AUC-ROC

These findings underscore the efficacy of pairing FOA with Random Forest for the detection of phishing websites, as demonstrated by both F-Measure and AUC-ROC metrics. This combination strikes a favorable balance between overall classification performance (AUC-ROC) and the precision-recall trade-off (F1-Measure). While the marginal discrepancies in AUC-ROC scores between algorithms imply that most approaches may suffice for phishing website detection, FOA with Random Forest appears to hold a slight advantage across both metrics.

5.5.4 Based on FPR

Figure 5 showcases the FPR scores for various combinations of ensemble classifiers and meta-heuristic feature selection algorithms as used in this study. FPR represents the proportion of legitimate websites incorrectly classified as phishing. The research findings are as follows:

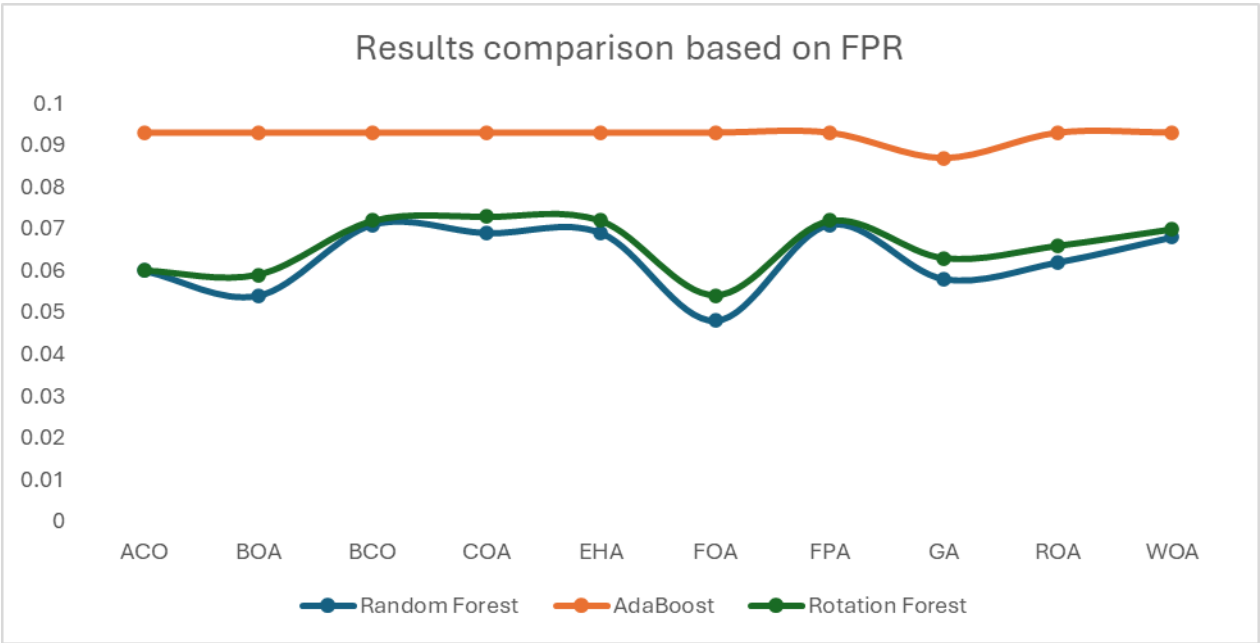


Figure 5: Results comparison based on FPR

- **Top Performer:** The combination of FOA for feature selection and Random Forest for classification achieves the lowest False Positive Rate (FPR) score of 0.048, indicating superior performance in minimizing the misclassification of legitimate websites as phishing.
- **Ensemble Classifier Performance:** Random Forest consistently outperforms other ensemble classifiers, with most FPR scores below 0.07. AdaBoost consistently exhibits the highest FPR across all feature selection algorithms, suggesting a tendency to misclassify legitimate websites as phishing.
- **Meta-heuristic Algorithm Performance:** FOA emerges as the top-performing meta-heuristic algorithm for feature selection based on FPR scores, consistently leading to lower false positive rates across all ensemble classifiers.
- **Overall Implications:** Combining FOA with Random Forest not only achieves strong overall performance and a balance between precision and recall but also minimizes the misclassification of legitimate websites as phishing. This is crucial for maintaining user experience, website traffic integrity and trustworthiness.
- **ACO and GA Performance:** ACO and GA demonstrate competitive performance across all metrics (F1-Measure, AUC-ROC, FPR). However, they are generally outperformed by FOA, suggesting FOA's slight superiority in selecting features for effective phishing website detection.
- **Strengths of ACO and GA:** ACO and GA are noted for their simplicity and ease of implementation compared to other meta-heuristic algorithms. However, they may not be as effective as FOA in exploring the search space for optimal feature combinations in this specific application.

5.6 Comparison of results with related studies

This section discusses the comparison between the proposed approach in this study and other related research in the literature. To provide transparent comparison with related studies, we compare the proposed approach in this study with those research that also focused on implementing feature selection algorithms since this is the main objective of this study. For clarity, each of these studies have been discussed in the related works section in order to show the achievement of each study. As shown in Table 6, the comparison is based on accuracy metric. This is because accuracy is the most common metrics across the different studies selected for comparison. In this section, the result of FOA and Random Forest as reported in this study is compared with other related studies to ascertain the overall performance of each model.

It can be seen that the proposed FOA with Random Forest achieved the best performance in terms of accuracy, showing the superiority of the proposed model in this study. The work of Adewole et al. (2021b) is based on Wolf optimization algorithm with Random Forest. Babagoli et al. (2019) implemented harmony search while Anupam and Kar (2021) considered Grey wolf optimization algorithm. Ali and Ahmed (2019) considered DNN-GA approach and achieve weighting accuracy of 91.13%. The results of the comparison with related studies showed that FOA combined with Random Forest produced the best overall performance. Therefore, the results positioned this combination as the best candidate for phishing website detection with minimal feature considerations.

Table 6: Results comparison with related studies

Models	Accuracy
Adewole et al. (2021b)	94.4%
Babagoli et al. (2019)	92.8%
Anupam and Kar (2021)	90.38%
Ali and Ahmed (2019)	91.13%
This paper	95.36%

6. Conclusion

Phishing attacks pose a significant threat to web-based applications, causing financial losses and compromising sensitive information. Traditional methods based on blacklists struggle to detect new (zero-day) attacks. This research addressed this limitation by exploring the efficacy of combining meta-heuristic algorithms for feature selection with ensemble classifiers for phishing website detection. The investigation of ten meta-heuristic algorithms revealed that Firefly Optimization Algorithm (FOA) achieved the most promising results in selecting a subset of features for phishing website detection.

Among the three ensemble classifiers evaluated (Random Forest, AdaBoost, Rotation Forest), Random Forest emerged as the strongest performer with an accuracy of 95.36% and AUC-ROC of 98.9%. This suggests that Random Forest, in combination with FOA-based feature selection, offers a robust approach for identifying phishing websites. Feature selection plays a crucial role in improving the performance of ensemble classifiers for phishing website detection. By selecting a relevant and informative subset of features, the overall classification accuracy and efficiency can be enhanced. This research highlights the potential of utilizing FOA for feature selection in conjunction with Random Forest classification for robust phishing website detection. By addressing the limitations of traditional methods, this approach offers a promising avenue for enhancing online security and protecting users from malicious attacks.

Contributions:

- This research provides a comprehensive analysis of ten meta-heuristic algorithms for feature selection in the context of phishing website detection.
- We demonstrate the effectiveness of combining FOA with ensemble classifiers like Random Forest for achieving superior performance in identifying phishing websites.
- Our findings contribute valuable insights to the ongoing research on developing robust and adaptable lightweight systems for combating phishing attacks.

Future Directions:

- Further exploration of other feature selection algorithms and their impact on the performance of ensemble classifiers could be beneficial.
- Analyzing the specific features chosen by FOA and how they contribute to accurate classification would provide deeper understanding of the model's effectiveness.
- Evaluating the proposed approach on real-world datasets collected over time would validate its generalizability and effectiveness against evolving phishing tactics.

Data availability: The dataset used in this study is publicly available for download on UCI dataset repository. The link to download the dataset has been mentioned in the dataset description section.

Conflict of interest: The authors declare that there is no conflict of interest.

Ethical approval: This research does not involve human or animal. However, it involved phishing website dataset which is publicly available.

Informed consent: There are no human subjects in this article and informed consent is not applicable.

References

- Adewole, K. S., Akintola, A. G., Salihu, S. A., Faruk, N., & Jimoh, R. G. (2019). *Hybrid rule-based model for phishing URLs detection*. Paper presented at the International Conference for Emerging Technologies in Computing.
- Adewole, K. S., Raheem, M. O., Abdulraheem, M., Oladipo, I. D., Balogun, A. O., & Baker, O. F. (2021a). Malicious URLs detection using data streaming algorithms. *Jurnal Teknologi dan Sistem Komputer*, 9(4), 224-229.
- Adewole, K. S., Raheem, M. O., Abikoye, O. C., Ajiboye, A. R., Oladele, T. O., Jimoh, M. K., & Aremu, D. R. (2021b). Malicious Uniform Resource Locator Detection Using Wolf Optimization Algorithm and Random Forest Classifier *Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics* (pp. 177-196): Springer.
- Adewole, K. S., Salau-Ibrahim, T. T., Imoize, A. L., Oladipo, I. D., AbdulRaheem, M., Awotunde, J. B., . . . Aro, T. O. (2022). Empirical Analysis of Data Streaming and Batch Learning Models for Network Intrusion Detection. *Electronics*, 11(19), 3109.
- Adil, M., Khan, R., & Ghani, M. A. N. U. (2020). *Preventive techniques of phishing attacks in networks*. Paper presented at the 2020 3rd International Conference on Advancements in Computational Sciences (ICACS).
- Alazaidah, R., Al-Shaikh, A., Al-Mousa, M., Khafajah, H., Samara, G., Alzyoud, M., . . . Almatameh, S. (2024). Website phishing detection using machine learning techniques. *Journal of Statistics Applications & Probability*, 13(1), 119-129.
- Ali, W., & Ahmed, A. A. (2019). Hybrid intelligent phishing website prediction using deep neural networks with genetic algorithm-based feature selection and weighting. *IET Information Security*, 13(6), 659-669.
- Anupam, S., & Kar, A. K. (2021). Phishing website detection using support vector machines and nature-inspired optimization algorithms. *Telecommunication Systems*, 76(1), 17-32.
- Asiri, S., Xiao, Y., Alzahrani, S., & Li, T. (2024). PhishingRTDS: A real-time detection system for phishing attacks using a Deep Learning model. *Computers & Security*, 141, 103843.
- Asiri, S., Xiao, Y., & Li, T. (2023a). PhishTransformer: A Novel Approach to Detect Phishing Attacks Using URL Collection and Transformer. *Electronics*, 13(1), 30.
- Asiri, S., Xiao, Y., Alzahrani, S., Li, S., & Li, T. (2023b). A survey of intelligent detection designs of HTML URL phishing attacks. *IEEE Access*, 11, 6421-6443.
- Babagoli, M., Aghababa, M. P., & Solouk, V. (2019). Heuristic nonlinear regression strategy for detecting phishing websites. *Soft Computing*, 23(12), 4315-4327.
- Balogun, A. O., Adewole, K. S., Raheem, M. O., Akande, O. N., Usman-Hamza, F. E., Mabayoje, M. A., . . . Jimoh, R. G. (2021). Improving the phishing website detection using empirical analysis of Function Tree and its variants. *Heliyon*, 7(7), e07437.
- Chiew, K. L., Tan, C. L., Wong, K., Yong, K. S., & Tiong, W. K. (2019). A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. *Information Sciences*, 484, 153-166.
- Cisco Talos Intelligence Group. (2023). Join the fight against phishing. Retrieved from <https://phishtank.org/>
- Google. (2023). Google Safe Browsing. Retrieved from <https://safebrowsing.google.com/>
- Karale, A. (2021). The challenges of IoT addressing security, ethics, privacy, and laws. *Internet of Things*, 15, 100420.

- Mosaad, M. I., abed el-Raouf, M. O., Al-Ahmar, M. A., & Banakher, F. A. (2019). Maximum power point tracking of PV system based cuckoo search algorithm; review and comparison. *Energy procedia*, 162, 117-126.
- Rondon, L. P., Babun, L., Aris, A., Akkaya, K., & Uluagac, A. S. (2022). Survey on enterprise Internet-of-Things systems (E-IoT): A security perspective. *Ad Hoc Networks*, 125, 102728.
- Soon, G. K., Chiang, L. C., On, C. K., Rusli, N. M., & Fun, T. S. (2020). *Comparison of ensemble simple feedforward neural network and deep learning neural network on phishing detection*. Paper presented at the Computational Science and Technology: 6th ICCST 2019, Kota Kinabalu, Malaysia, 29-30 August 2019.
- Taofoek, A. O. (2024). Development of a Novel Approach to Phishing Detection Using Machine Learning. *ATBU Journal of Science, Technology and Education*, 12(2), 336-351.
- URIBL. (2023). Public url blacklist Retrieved from <https://uribl.com/>
- van Geest, R., Cascavilla, G., Hulstijn, J., & Zannone, N. (2024). The applicability of a hybrid framework for automated phishing detection. *Computers & Security*, 139, 103736.
- Vitorino, J., Andrade, R., Praça, I., Sousa, O., & Maia, E. (2022). *A comparative analysis of machine learning techniques for iot intrusion detection*. Paper presented at the International Symposium on Foundations and Practice of Security.
- Vrbančič, G., Fister Jr, I., & Podgorelec, V. (2018). *Swarm intelligence approaches for parameter setting of deep learning neural network: case study on phishing websites classification*. Paper presented at the Proceedings of the 8th international conference on web intelligence, mining and semantics.
- Wang, G.-G., Deb, S., Gao, X.-Z., & Coelho, L. D. S. (2016). A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. *International Journal of Bio-Inspired Computation*, 8(6), 394-409.
- Wei, B., Hamad, R. A., Yang, L., He, X., Wang, H., Gao, B., & Woo, W. L. (2019). A deep-learning-driven light-weight phishing detection sensor. *Sensors*, 19(19), 4258.
- Yang, P., Zhao, G., & Zeng, P. (2019). Phishing website detection based on multidimensional features driven by deep learning. *IEEE Access*, 7, 15196-15209.
- Yang, X.-S. (2009). *Firefly algorithms for multimodal optimization*. Paper presented at the Stochastic Algorithms: Foundations and Applications: 5th International Symposium, SAGA 2009, Sapporo, Japan, October 26-28, 2009. Proceedings 5.
- Zamir, A., Khan, H. U., Iqbal, T., Yousaf, N., Aslam, F., Anjum, A., & Hamdani, M. (2020). Phishing web site detection using diverse machine learning algorithms. *The Electronic Library*, 38(1), 65-80.
- Zhu, E., Ju, Y., Chen, Z., Liu, F., & Fang, X. (2020). DTOF-ANN: an artificial neural network phishing detection model based on decision tree and optimal features. *Applied soft computing*, 95, 106505.
- Zscaler. (2022). 2022 Zscaler ThreatLabz - State of Phishing Report. *Industry Report*. Retrieved from <https://info.zscaler.com/>