

A Hybrid Metaheuristic Feature Selection Framework for Enhanced Phishing Detection

Abdallah S. Hyassat^{1*}, Sumaya Al-Khatib², Motasem M. Elshourbagy³, Mosleh M. Abualhaj⁴,
Qusai Y. Shambour⁵, Mohammad Sh. Daoud^{6*}, Mohammad A. Alsharaiah⁷, Mohammed Anbar⁸

Faculty of Information Technology-Data Science and Artificial Intelligence Department,
Al-Ahliyya Amman University, Amman, Jordan¹

Department of Computer Science-Faculty of Information Technology, The University of Jordan, Amman, Jordan¹

Faculty of Information Technology-Networks and Cybersecurity Department,

Al-Ahliyya Amman University, Amman, Jordan^{2, 3, 4}

Department of Physics and Engineering Mathematics-Mattaria Faculty of Engineering, Helwan University, Egypt³

Faculty of Information Technology-Software Engineering Department, Middle East University, Amman, Jordan⁵

College of Engineering, Al Ain University, Abu Dhabi, United Arab Emirates⁶

Faculty of King Abdullah II School of Information Technology-Information Technology Department,
The Jordan University, Amman, Jordan⁷

Cybersecurity Research Center (CYRES), Universiti Sains Malaysia (USM), Penang, Malaysia⁸

Abstract—Phishing websites remain a critical cybersecurity threat due to their ability to mimic legitimate platforms and deceive users into revealing sensitive information. This study proposes a phishing detection framework that centers on a novel union-based feature selection strategy, WOA union DA, which combines the Whale Optimization Algorithm (WOA) and Dragonfly Algorithm (DA). Instead of relying on a single metaheuristic, the proposed union method merges the feature subsets selected by both algorithms to capture complementary search patterns and reduce selection bias. This unified subset enhances diversity, preserves informative attributes, and improves generalization. Evaluated on the ISCX-URL2016 dataset containing 15,358 labeled URLs, the proposed approach is integrated with XGBoost and SVM classifiers. Experimental results show that WOAUDA achieves 97.95% accuracy with XGBoost and 96.29% with SVM, representing modest improvements over the individual WOA and DA methods. Comparative analysis with existing models further demonstrates the competitive classification performance of the proposed approach.

Keywords—Phishing; machine learning; feature selection; whale optimization algorithm; dragonfly algorithm

I. INTRODUCTION

The Internet has become an essential part of daily life, connecting people across the world and supporting many important services such as communication, online business, cloud platforms, and data exchange [1], [2], [3]. It plays a central role in key sectors including healthcare, finance, education, and government services. Because these areas depend heavily on online systems, the reliability and availability of Internet services are vital for economic development and social stability [4], [5], [6]. As societies continue to rely more on digital technologies, ensuring the security of these services has become a major challenge and an urgent priority [7], [8], [9].

Web browsers are one of the primary tools people use to access Internet services. They are software programs people use to access and interact with websites. A website is a collection of web pages hosted on a server and accessible through a domain name. Websites provide almost all the services that the Internet offers to users. These services include searching for information, making a payment on an e-commerce site, booking a flight or hotel online, etc. [10], [11]. As of December 2023, there were about 1.1 billion websites worldwide, and by January 2025 the number of indexable websites reached approximately 3.76 billion [12], [13]. The huge web usage makes it very attractive for cybercriminals. Cybercriminals are using websites to launch different types of attacks on users and businesses. These attacks include malware distribution, SQL injection, cross-site scripting, credential stuffing, phishing attacks, etc. [14], [15], [16].

Phishing attacks involve fraudulent websites or login interfaces designed to deceive users into revealing sensitive information. Fig. 1 presents the workflow of a typical phishing attack. Common types of phishing attacks are Pharming, Watering hole phishing, and email phishing [17], [18], [19]. Phishing was the costliest attack vector in 2022, with each data breach averaging USD 4.91 million [20], [21]. This is an alarming number indicating a rise in phishing attacks and their significant financial impact. For instance, credential theft and financial loss occurred in 2023 by sending fraudulent invoices from genuine PayPal domains [22].

Machine Learning (ML) has been widely used over the years to alleviate phishing attacks. ML techniques operate by analyzing historical phishing attack data and building models that can detect new or evolving attacks. ML has proven to be a very efficient science that has been adopted in several domains, including cybersecurity [23], [24], [25]. However, ML faces several challenges that hinder its performance in detecting phishing attacks. One such challenge is dealing with high-dimensional data, which contains a large number of phishing-

*Corresponding author.

related features. These data increase the processing time and reduce the accuracy of the ML-based phishing attack detection model [26], [27]. Generally, data features can be classified into relevant features and irrelevant features. Removing irrelevant features will enhance phishing attack detection (increase accuracy and reduce processing time). Typically, ML models use Feature Selection (FS) algorithms, including metaheuristic algorithms, to remove irrelevant features [28], [29].

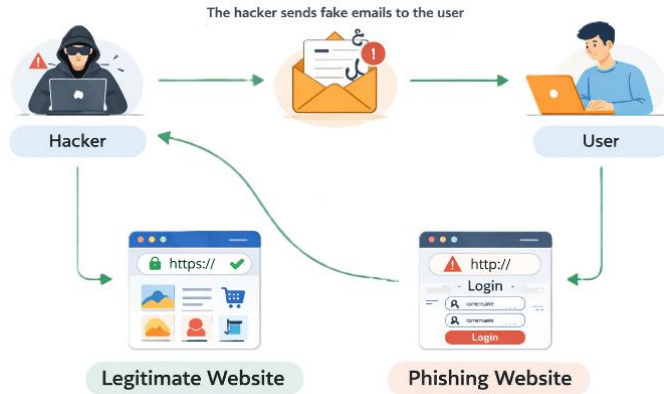


Fig. 1. Phishing workflow.

In problems where finding an exact solution is computationally infeasible, metaheuristic algorithms are used to find near-optimal solutions. Metaheuristic algorithms apply certain rules with controlled randomness to search diverse areas of the solution space and focus on promising areas. Examples of such algorithms include the Harris Hawks Optimizer, Bat Algorithm, Numbat Optimization Algorithm (NOA), Tuatara Optimization Algorithm (TOA), Solenodon Optimization Algorithm (SOA), Whale Optimization Algorithm (WOA), and Dragonfly Algorithm (DA). In this study, the WOA and DA algorithms will be used for feature selection [23], [30], [31], [32], [33], [34]. In this study, the WOA and DA algorithms will be used for feature selection. Particularly, they will be used to search for the features of the phishing attack and remove irrelevant features to keep only the nearly optimal features that can be used to identify phishing attacks. In addition, several ML algorithms will be customized

to classify the phishing attacks accurately. Therefore, the key contributions of this study are as follows: 1) Developing an ML model specifically designed to detect phishing attacks, 2) Applying the WOA and DA metaheuristic algorithms to identify key features relevant to phishing detection, 3) Customizing the ML algorithms by selecting optimal hyperparameters using the Random Search (RS) algorithm to enhance their performance in phishing attack detection, and 4) Evaluating the performance of the proposed ML model and comparing it with existing models.

II. BACKGROUND

This section provides the necessary background to understand the proposed model. It reviews key ML algorithms and techniques relevant to phishing detection. The aim is to establish a foundation for the methods and design choices discussed later in the study.

A. ISCX-URL2016 Phishing Dataset

The ISCX-URL2016 dataset, developed by the Canadian Institute for Cybersecurity at the University of New Brunswick, contains several URL categories, including benign, phishing, malware, spam, and defacement. This study focuses on a binary phishing detection subset containing 15,358 labeled URLs, comprising 7,577 benign and 7,781 phishing instances. All the records are characterized by 79 features, which represent different URL features such as lexical, domain-based, and content-related features. There is noise and missing values in some features, and therefore data cleaning is required. Some features are numerically different and need to be normalized before the model is trained. Table I summarizes the feature names, descriptions, type, minimum, and maximum values. The dataset represents actual patterns and techniques of phishing attacks in real life and assists in determining malicious intent in URL structure and metadata. In addition, phishing URLs impersonate trusted sites to defraud users and, in many cases, deceive users through URL obfuscation. Moreover, it enables the assessment of the classifiers and feature selection techniques and assists in real-time studies of phishing detection [35], [36].

TABLE I. SUMMARY OF THE DATASET FEATURES

#	Feature Name	Description	Min. Value	Max. Value	#	Feature Name	Description	Min. Value	Max. Value
1	Querylength	Total length of the query string (after '?')	0	173	2	domain_token_count	Number of tokens in the domain (split by dots or hyphens)	2	19
3	path_token_count	Number of tokens in the path (split by '/')	0	68	4	avgdomaintokenlength	Average length of tokens in the domain	1.5	29.5
5	longdomaintokenlength	Longest token length in the domain	2	63	6	avgpathtokenlength	Average length of tokens in the path	0	105
7	tld	Count of top-level domain elements (e.g., .com, .co.uk)	2	19	8	charcompvowels	Ratio of vowels to total characters in URL	0	94
9	charcomppace	Ratio of space-like characters (e.g., %20) in URL	0	62	10	ldl_url	Frequency of letter-digit-letter pattern in full URL	0	58
11	ldl_domain	Same pattern frequency in the domain	0	37	12	ldl_path	Same pattern frequency in path	0	46
13	ldl_filename	Same pattern frequency in file name	0	31	14	ldl_getArg	Same pattern frequency in query argument values	0	39
15	dld_url	Digit-letter-digit pattern frequency in full URL	0	38	16	dld_domain	Same pattern frequency in domain	0	25

#	Feature Name	Description	Min. Value	Max. Value	#	Feature Name	Description	Min. Value	Max. Value
17	dld_path	Same pattern frequency in path	0	32	18	dld_filename	Same pattern frequency in file name	0	14
19	dld_getArg	Same pattern frequency in argument values	0	31	20	urlLen	Total length of the full URL	37	348
21	domainlength	Total length of the domain	4	227	22	pathLength	Length of the path section of the URL	0	330
23	subDirLen	Length of the sub-directory path (excluding filename)	0	330	24	fileNameLen	Length of the file name in the URL	1	216
25	this.fileExtLen	Length of the file extension (e.g., .php, .html)	1	5	26	ArgLen	Length of the arguments/query string	0	279
27	pathurlRatio	Ratio of path length to full URL length	0	0.95086706	28	ArgUrlRatio	Ratio of argument length to full URL length	0	0.9109589
29	argDomanRatio	Ratio of argument length to domain length	0	20.461538	30	domainUrlRatio	Ratio of domain length to URL length	0.028901733	0.94520545
31	pathDomainRatio	Ratio of path length to domain length	0	32.9	32	argPathRatio	Ratio of argument length to path length	0	2
33	executable	Binary flag: whether the URL points to an executable file	0	1	34	isPortEighty	Binary flag: whether port 80 is explicitly included	-1	0
35	NumberOfDotsinURL	Total number of "." in the URL	1	20	36	ISIpAddressInDomainName	Binary flag: if domain is an IP address	-1	-1
37	CharacterContinuityRate	Ratio of longest runs of same character type to URL length	0.07514451	1	38	LongestVariableValue	Longest value found in the query parameters	-1	164
39	URL_DigitCount	Number of digits in the full URL	0	144	40	host_DigitCount	Number of digits in the domain	0	64
41	Directory_DigitCount	Number of digits in the path section	-1	86	42	File_name_DigitCount	Number of digits in the file name	-1	91
43	Extension_DigitCount	Number of digits in the extension	-1	91	44	Query_DigitCount	Number of digits in the query string	-1	87
45	URL_Letter_Count	Number of letters in the full URL	11	290	46	host_letter_count	Number of letters in the domain	2	189
47	Directory_LetterCount	Number of letters in directory	-1	183	48	Filename_LetterCount	Number of letters in file name	-1	153
49	Extension_LetterCount	Number of letters in extension	-1	261	50	Query_LetterCount	Number of letters in query string	-1	124
51	LongestPathTokenLength	Length of the longest token in path	0	304	52	Domain_LongestWordLength	Longest word/token in the domain	2	40
53	Path_LongestWordLength	Longest word/token in the path	0	99	54	sub-Directory_LongestWordLength	Longest token in subdirectory (path)	-1	48
55	Arguments_LongestWordLength	Longest token in query argument values	-1	91	56	URL_sensitiveWord	Count of sensitive words (e.g., "login", "bank") in URL	0	3
57	URLQueries_variable	Number of query parameters (key=value pairs)	0	12	58	spcharUrl	Count of special characters in URL (e.g., @, %, &)	1	17
59	delimiter_Domain	Count of delimiters in domain (e.g., -, .)	0	10	60	delimiter_path	Count of delimiters in path (e.g., /, -)	0	64
61	delimiter_Count	Total number of delimiters in full URL	-1	23	62	NumberRate_URL	Digit-to-length ratio of the full URL	0	0.761904762
63	NumberRate_Domain	Digit-to-length ratio of domain	0	0.666666667	64	NumberRate_DirectoryName	Digit-to-length ratio of directory	-1	0.935064935
65	NumberRate_FileName	Digit-to-length ratio of file name	-1	1	66	NumberRate_Extension	Digit-to-length ratio of file extension	-1	1
67	NumberRate_AfterPath	Digit-to-length ratio of everything after the path	-1	1	68	SymbolCount_URL	Count of symbols in the full URL	2	47
69	SymbolCount_Domain	Count of symbols in the domain	1	18	70	SymbolCount_Directoryname	Count of symbols in directory name	-1	24
71	SymbolCount_FileName	Count of symbols in file name	-1	31	72	SymbolCount_Extension	Count of symbols in extension	-1	30
73	SymbolCount_Afterpath	Count of symbols after the path section	-1	29	74	Entropy_URL	Shannon entropy of full URL	0.419559536	0.869700977
75	Entropy_Domain	Shannon entropy of domain	0.561912714	1	76	Entropy_DirectoryName	Shannon entropy of directory	-1	0.962478638
77	Entropy_Filename	Shannon entropy of file name	-1	1	78	Entropy_Extension	Shannon entropy of extension	-1	1
79	Entropy_Afterpath	Shannon entropy of URL part after path	-1	1					

B. Nature-Inspired Algorithms for Feature Selection

Swarm intelligence-based methods such as the WOA and the DA algorithms have shown promising results in feature selection tasks. This section introduces the fundamental concepts of WOA and DA, followed by their application in selecting relevant features to improve the performance of phishing attack detection models.

1) *WOA Algorithm*: The WOA algorithm mimics bubble-net hunting behavior to balance exploration and exploitation through encircling, spiral movement, and adaptive random search mechanisms. Exploration is strengthened using random whale positions and adaptive coefficients [Eq. (1) and (2)], while exploitation focuses on the current best solution via a spiral convergence model with stochastic parameter tuning [Eq. (3)-(5)]. Feature subsets are encoded as binary vectors using transfer functions that map continuous updates to binary selections, and the fitness function balances classification accuracy with feature subset size. WOA maintains search diversity through random movements and coefficient variation, converges rapidly with low computational cost, but may be prone to local optima depending on the coefficient A [Eq. (1)] and the adaptive parameter a [Eq. (6)]. Algorithm 1 illustrates the WOA pseudocode [17], [23], [37].

$$A = 2a r - a \quad (1)$$

$$C = 2r \quad (2)$$

$$D = |C \cdot X^*(t) - X(t)| \quad (3)$$

$$X(t+1) = X^*(t) - A \cdot D \quad (4)$$

$$X(t+1) = D' \cdot e^{bl} \cos(2\pi l) + X^*(t) \quad (5)$$

$$a = 2 - \left(\frac{2t}{T}\right) \quad (6)$$

where, $X(t)$ represents the current whale position at iteration t , $X^*(t)$ denotes the best solution found so far, A and C are coefficient vectors controlling the exploration and exploitation behavior, D is the distance between the whale and the prey (best solution), D' represents the distance used in the spiral movement, b is a constant defining the spiral shape, l is a random number in the range $[-1, 1]$, r is a random number in $[0, 1]$, a is a control parameter that decreases linearly during the search process, t denotes the current iteration, and T represents the maximum number of iterations.

Algorithm 1: Whale Optimization Algorithm (WOA)

```
1: Initialize
2:   Generate an initial population of whales with random positions.
3:   Evaluate the fitness of each whale.
4:   Identify the whale with the best fitness as globalBest
5:   for each iteration do
6:     Update parameters  $a$  linearly decreasing from 2 to 0.
7:     for each whale in the population do
8:       Generate two random numbers  $r \in [0, 1]$ .
9:       if  $r < 0.5$  then
10:        Exploitation (encircling the best whale):
```

```
11:         newPosition  $\leftarrow$  globalBest.position -  $r \times$ 
12:           |globalBest.position - position|
13:       else if  $r \geq 0.5$  then
14:         Local exploitation (search near a random whale):
15:         Select a random whale  $w_j$ 
16:         newPosition  $\leftarrow w_j$ .position -  $r \times |w_j$ .position -
17:           current.position|
18:       else
19:         Global exploration (spiral updating position):
20:         Generate a random number  $p \in [0, 1]$ .
21:         newPosition  $\leftarrow$  globalBest.position -
22:           current.position  $\times (a \times r)$ 
23:       end if
24:       Apply boundary handling to ensure newPosition is
25:       within the valid search space.
26:       Update the current whale's position with newPosition
27:     end for
28:   Evaluate the fitness of all updated whales
29:   Update globalBest if a better solution is found.
30: end for
31: Return globalBest as the final selected feature subset
```

2) *DA Algorithm*: The DA algorithm simulates static and dynamic swarming behaviors for local and global search. It explores the feature space using alignment, cohesion, and separation among swarm members [Eq. (7)-(9)], and exploits local solutions via attraction to optimal neighbors (food sources) [Eq. (10)] while avoiding unfavorable regions through enemy distraction [Eq. (11)]. Feature selections are updated using multi-force behavioral dynamics, relying on swarm-based operators like repulsion, attraction, and neighborhood interaction through velocity and position updates [Eq. (12) and (13)]. Features are encoded in binary vectors through transfer function discretization, while the fitness function penalizes large subsets and rewards classification accuracy [34], [38]. DA introduces randomness through behavior weights and a dynamic neighborhood radius, which helps maintain search diversity via dynamic behavior switching. Though it converges more slowly than WOA, it better avoids premature convergence and is more robust in escaping local optima due to swarm intelligence. The algorithm is sensitive to swarm radius and the balance of interaction forces, and is computationally heavier due to complex force calculations. Algorithm 2 presents the pseudocode of the DA algorithm [34], [38].

$$S_i = \sum_{j=1}^n (X_i - X_j) \quad (7)$$

$$A_i = \frac{1}{N} \sum_{j=1}^N V_j \quad (8)$$

$$C_i = \left(\frac{1}{N} \sum_{j=1}^N X_j \right) - X_i \quad (9)$$

$$F_i = X^+ - X_i \quad (10)$$

$$E_i = X^- + X_i \quad (11)$$

$$V_i(t+1) = W V_i(t) + s S_i + a A_i + c C_i + f F_i + e E_i \quad (12)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (13)$$

where, X_i represents the position of the i^{th} dragonfly, V_i denotes its velocity vector, X_j and V_j represent the position and velocity of neighboring dragonflies, N is the number of neighbors, S_i is the separation force, A_i is the alignment force, C_i is the cohesion force, F_i represents attraction toward the food source X^+ , E_i represents distraction from the enemy X^- , w is the inertia weight controlling momentum, and s , a , c , f , and e are the corresponding behavioral weights controlling the influence of separation, alignment, cohesion, food attraction, and enemy distraction forces.

Algorithm 2: Dragonfly Algorithm (DA)

```
1: Initialization
2:   for each dragonfly df in the population do
3:     Assign a random position vector psn
4:     Assign a random velocity vector vel
5:   end for
6: Evaluate the fitness of each dragonfly
7: Identify the dragonfly with the best fitness as globalBest
8: for each iteration do
9:   Update inertia weight w (typically decreases linearly)
10:  for each dragonfly in the population do
11:    Compute behavioral components based on neighbors:
12:    Separation (S): steer away from nearby individuals
13:    Alignment (A): align velocity with nearby individuals
14:    Cohesion (C): move toward the center of neighbors
15:    Attraction to food (F): move toward globalBest
16:    Distraction from enemy (E): move away from worst solution
17:    Update velocity:
18:      vel = w × vel + S + A + C + F + E
19:    Update position
20:      psn = psn + vel
21:    Apply boundary constraints to keep psn within search space
22:  end for
23: Evaluate fitness of all updated dragonflies
24: Update globalBest if a better solution is found
25: end for
26: Return globalBest as the final selected feature subset.
```

3) *Feature selection with DA and WOA algorithms*: DA and WOA algorithms improve classification accuracy through optimal feature selection. In addition, they reduce dimensionality by removing irrelevant and redundant URL features. By eliminating noisy features, they further enhance model generalization. Both algorithms are capable of handling high-dimensional phishing datasets effectively and reducing both lexical and structural phishing URL features. Moreover, they retain key patterns crucial for phishing detection and enhance the detection of obfuscated and evolving phishing URLs. They also support early-stage phishing threat identification and produce compact and interpretable feature subsets. Furthermore, they adapt to different phishing attack strategies while reducing classifier training time and memory

usage. These methods also lower computational costs for large-scale phishing datasets and exhibit fast convergence that is suitable for real-time detection tasks. DA, in particular, can be integrated but may be computationally heavier depending on the implementation. Overall, both DA and WOA show strong benchmarking performance in phishing detection tasks. Table II shows the main properties of the DA and WOA algorithms [34], [37], [39], [40], [41].

TABLE II. MAIN PROPERTIES OF DA AND WOA ALGORITHMS

Property	DA Algorithm	WOA Algorithm
Nature-Inspired Behavior	Swarming behavior of dragonflies	Hunting strategy of humpback whales
Search Style	Mimics swarm movement and group interactions	Mimics encircling and spiral motion around prey
Main Strength	Thorough exploration and noise tolerance	Fast convergence and simplicity
Feature Selection Style	Picks diverse and supporting features	Picks minimal and strong features
Speed	Slower but stable	Faster and lightweight
Complexity	More complex (many behaviors to manage)	Simple and easy to implement
Best For	Noisy or high-dimensional datasets	Clean or well-separated datasets
Overfitting Risk	Low	Higher if subset is too small
Generalization Ability	Strong (handles varied patterns)	Moderate
Suitability for Real-Time	Less ideal without optimization	Well-suited

C. ML Classifiers used in the Proposed Model

This section describes the machine-learning classifiers used in the proposed phishing detection model. Specifically, SVM and XGBoost are employed due to their proven effectiveness in classification tasks.

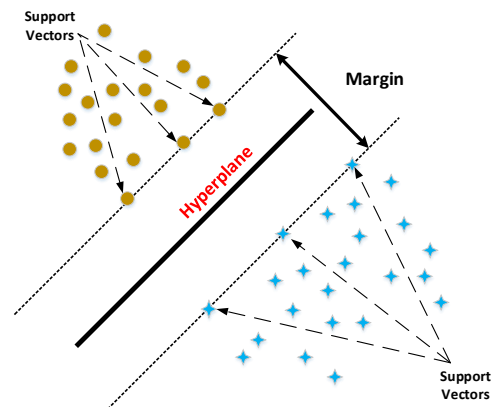


Fig. 2. SVM classifier scheme.

1) *SVM Classifier*: Finding the appropriate separating hyperplane that optimizes the margin between the classes is the fundamental principle of SVMs. In a two-dimensional space, the margin is the distance between the separating hyperplane and the nearest data points of each class. The hyperplane that maximizes the margin is chosen as the decision boundary. The data points closest to the separating

hyperplane are called support vectors. These points play a crucial role in determining the hyperplane and the margin. Fig. 2 clarifies the SVM technique. SVMs can be sensitive to the choice of kernel function, and the training time can be long for large datasets. The kernel function that will be used with SVM in the proposed system is the radial basis function, because it can handle high-dimensional data and non-linear relationships between features [42], [43], [44].

2) *XGBoost classifier*: XGBoost builds sequential decision trees to improve classification accuracy by using gradient boosting to minimize the classification loss function. The model starts with a base prediction and calculates gradients and Hessians to measure errors. It then builds trees by splitting data to minimize loss, using gain scores to select optimal splits, and pruning low-gain splits. Eq. (14) shows the final prediction, which is a weighted sum of the predictions from all trees [45], [46], [47].

$$\hat{y}(x) = \sum_{m=1}^M f_m(x) \quad (14)$$

where, $\hat{y}(x)$ is the raw prediction, M is the number of trees, and $f_m(x)$ is the prediction from the m -th tree. XGBoost corrects errors from previous trees, applying a learning rate to scale their contributions and using L1/L2 regularization to prevent overfitting. It also handles missing data by learning the best default directions. For binary classification, this raw output is passed through a sigmoid function to convert it into a probability. Table III compares and contrasts the SVM and XGBoost classifiers [45], [46], [47].

TABLE III. SVM AND XGBOOST CLASSIFIERS

Aspect	SVM	XGBoost
Model Type	Margin-based classifier	Optimized ensemble of decision trees
Algorithmic Foundation	Optimization of margin between classes	Regularized gradient boosting (CART-based)
Loss Function	Hinge loss (or its variants)	Log loss + regularization (L1, L2)
Regularization Support	Soft margin with C parameter	Explicit L1 and L2 regularization
Handling of Linearly Separable Data	Optimal with linear kernel	Effective, but more complex than needed
Hyperparameter Tuning Complexity	High (especially with RBF kernel)	High (many interdependent parameters)
Robustness to Irrelevant Features	Low to moderate	High (feature splitting reduces noise impact)

D. RS Algorithm

The RS algorithm randomly selects combinations from predefined hyperparameter distributions and explores a wide range of parameter values without testing all possible combinations. It is efficient for high-dimensional hyperparameter spaces and performs well when only a few parameters have a significant influence on model performance. Each iteration evaluates a new random configuration, and the process is easily parallelizable due to the independence of model evaluations. It selects the best-performing set based on evaluation scores and helps avoid local optima through random sampling. RS is unbiased toward any specific parameter value,

does not assume relationships between parameters, and works with any machine learning classifier or regressor. Algorithm 3 presents the pseudocode of the RS algorithm [48], [49].

Algorithm 3: Random Search (RS)

Require: Machine learning model M , hyperparameter search space H , number of evaluations N , training dataset D_{train} ,

Ensure: Best hyperparameter configuration H_{best} , best evaluation score S_{best}

```

1: Initialize  $S_{\text{best}} \leftarrow 0$ 
2: Initialize  $H_{\text{best}} \leftarrow \emptyset$ 
3: for  $i = 1$  to  $N$  do
4:   Sample  $h_i \in H$  by randomly selecting one value
5:   for each hyperparameter in model  $M$  ( $h_i$ )
6:     Evaluate  $M(h_i)$  on  $D_{\text{val}}$  using  $\text{Eval}(\cdot) \rightarrow \text{score}_i$ 
7:     if  $\text{score}_i > S_{\text{best}}$  then
8:        $S_{\text{best}} \leftarrow \text{score}_i$ 
9:        $H_{\text{best}} \leftarrow h_i$ 
10:    end if
11:  end for
12: Return  $H_{\text{best}}, S_{\text{best}}$ 

```

III. RELATED WORKS

M. K. Prabakaran et al. [50] present a deep learning-based approach for phishing detection by proposing a hybrid framework that integrates a Variational Autoencoder (VAE) with a Deep Neural Network (DNN). In the proposed architecture, URL strings are first transformed into fixed-size one-hot encoded matrices of dimension 116×84 to ensure consistent input formatting. The VAE is employed as an automated feature learning module, mapping raw URL representations into compact latent vectors. Particular attention is given to optimizing the latent space, as appropriate dimensionality is shown to enhance discrimination capability while simultaneously reducing redundancy and noise. Empirical analysis indicates that a latent dimension of 24 yields the lowest reconstruction loss during VAE training. The extracted latent features are subsequently fed into a DNN classifier to determine whether a given URL is malicious or benign. Evaluation is conducted using the publicly available ISCX-URL2016 phishing dataset with an 80/20 training-testing split. The proposed method achieves an accuracy of 97.45%, surpassing several comparative deep learning models, including CNN, Vanilla AE, Sparse AE, and Denoising AE.

Vinayakumar R. et al. [51] proposed a DL model to identify malicious phishing URLs that are not detected by traditional blacklist-based systems. Their algorithm is called DeepURLDetect (DUD), and it is a hybrid architecture that combines CNN and Long Short-Term Memory (LSTM). It is designed to encode raw URLs on a character-by-character basis with Keras embeddings to retain sequential features in URLs and does not require any human feature engineering. The CNN layer was responsible for retrieving local features (substrings and common phishing tokens, e.g., login, PayPal), whereas the LSTM layer was used to gather long-term dependencies and sequential linkages between characters of the URLs. The DUD model uses a sigmoid activation function to classify binary (malicious vs. benign) and learns adaptively by

reducing the binary cross-entropy loss with the Adam optimizer. The results of Dataset1 (D1), Dataset2-Random split (D2R), and Dataset2-Time split (D2T) showed accuracies of 97.2%, 95.4%, and 93.1%, respectively. The 3-gram DNN with a 5-layer baseline model also performed well, with accuracies of 95.4%, 95%, and 93% across the same datasets, respectively.

E. Gandotra and D. Gupta [52] emphasize the impact of selecting relevant features on the performance and efficiency of ML models. They built a phishing website dataset from two public sources. The resulting dataset includes 30 features and labeled instances of both phishing and legitimate websites. The study aims to maintain high accuracy by using a smaller yet more informative feature set. Feature selection is performed using the IG method, combined with a Ranker algorithm, to identify the top 15 features. The RF classifier, along with other models, is evaluated using a 10-fold cross-validation approach. RF achieves the highest accuracy of 96.3% after feature selection. Therefore, the selected 15 features preserve strong classification performance.

SJ. Bu and HJ. Kim [53] have suggested a hybrid framework of DL and Genetic Algorithms (GA) to improve phishing website detection. The GA-enhanced DL model specifically focuses on phishing URLs that were able to circumvent conventional detection filters. The model uses the CNN-LSTM model to extract character-level, as well as word-level, features of URLs. The CNN layers in this architecture detect local patterns from suspicious substrings using recall and cross-validation functions. LSTM layers captured the

sequential dependencies in URL components such as domain structures. The GA algorithms were used to find the best combinations of features, whereby the objective was to reduce false negatives. A recall function was applied as the fitness in the GA with emphasis on the sensitivity of detection. The cross-validation function was used to optimize the model architecture and the hyperparameters, and feature importance was identified through SHAP values and visualized through different techniques. The three datasets were analysed using CNNLSTM, ISCX-URL-2016, PhishStorm, and PhishTank. The results showed accuracies of 96.85%, 95.05%, and 94.83% on the different datasets, respectively.

B. Geyik et al. [54] investigate phishing threats that rely on misleading URLs to obtain confidential user data. Their work focuses on identifying phishing websites through machine-learning-based classification of URL-derived attributes. Experiments are conducted on the CatchPhish_D3 dataset, which contains 122,055 URLs labeled as either phishing or benign. The URLs are parsed into their constituent components, and these elements are transformed into structured numerical representations suitable for machine learning. In total, 15 lexical and structural attributes are derived from each URL—such as protocol type, hostname length, path characteristics, and delimiter counts—forming the core preprocessing stage. Four machine learning classifiers are evaluated, with the Random Forest model demonstrating the best performance, achieving an accuracy of 83%. Table IV provides a summary of existing approaches to phishing attack detection.

TABLE IV. SUMMARY OF EXISTING APPROACHES FOR PHISHING ATTACK DETECTION

Reference	Model	Feature Reduction Method	Dataset Name	Results (Accuracy)
M. K. Prabakaran et al. [50]	DNN	VAE	ISCX-URL2016 datasets	97.45%
	CNN			93.18%
	Vanilla AE			94.73%
	Sparse AE			95.09%
	Denoising AE			95.71%
Vinayakumar R. et al. [51]	DUD-D1	Implicitly using CNN and LSTM	Dataset1 (built from multiple public datasets) and Dataset2 (private from Sophos Labs)	97.2%
	DUD-D2R			93.1%
	DUD-D2T			93.1%
	3-gram-D1	Implicitly using DNN		95.4%
	3-gram-D2R			95%
	3-gram-D2T			93%
E. Gandotra & D. Gupta [52]	RF	IG	Built from two public sources	96.3%
SJ. Bu and HJ. Kim [53]	CNN-LSTM	GA algorithm	ISCX-URL-2016	96.85%
B. Geyik et al. [54]	RF	No feature reduction	CatchPhish_D3	83%

IV. PHISHING DETECTION MODEL

In this section, we will discuss the proposed phishing detection model. The framework starts by processing the ISCX-URL2016 phishing dataset. Then, the proposed feature selection method will be deployed to eliminate the irrelevant features. Finally, the produced subset of features will be

evaluated using several ML classifiers. Fig. 3 clarifies the proposed phishing detection model.

A. URL-Phishing Dataset Preprocessing

Preprocessing converts raw data into a clean, usable format. It prepares data for training robust ML models and addresses missing values, outliers, and inconsistent scales. Raw data often contains missing or invalid entries, and unprocessed data

can mislead training algorithms. Feature scales vary and may bias certain models, while noise or extreme values reduce

prediction accuracy. In contrast, clean and normalized data improve learning efficiency and stability [55], [56].

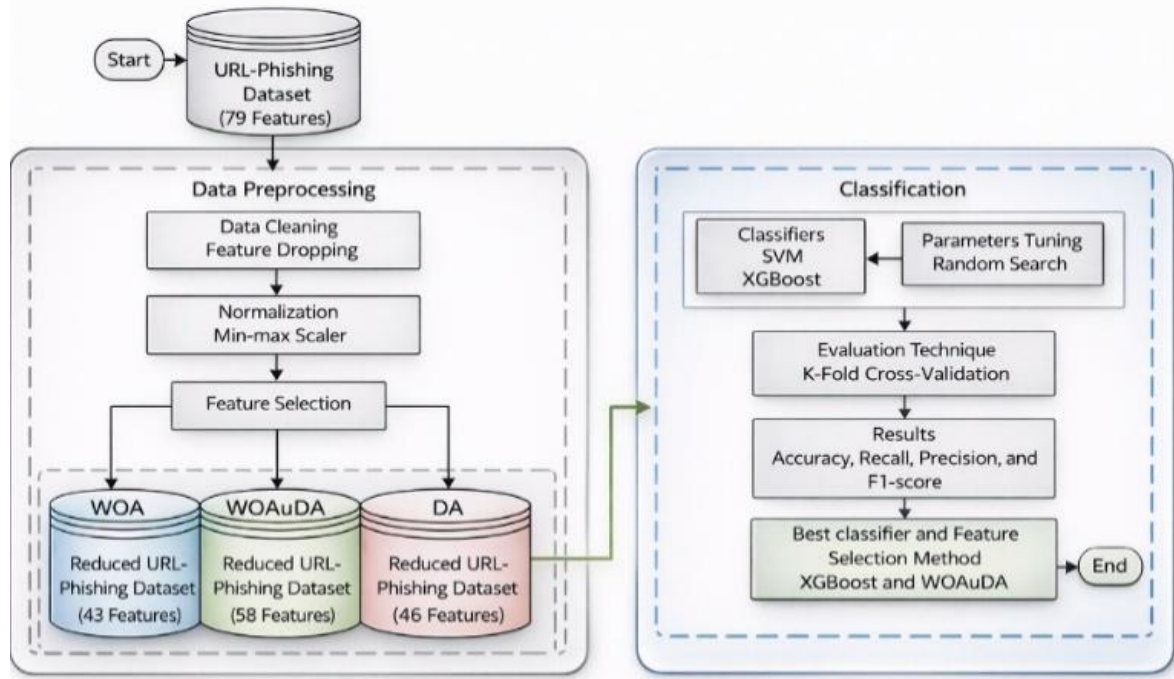


Fig. 3. The proposed phishing detection model.

The preprocessing phase begins by loading the ISCX-URL2016 phishing dataset and inspecting its size and data types. Statistical properties of each feature, including minimum, maximum, mean, and distribution, are then analyzed, followed by checks for missing, infinite, or duplicate values. In this dataset, NaN values were detected in nine features, while one feature contained infinite values, namely avgpathtokenlen, argPathRatio, NumberRate_DirectoryName, NumberRate_FileName, NumberRate_Extension, NumberRate_AfterPath, Entropy_DirectoryName, Entropy_FileName, Entropy_Extension, and Entropy_AfterPath. To maintain data integrity and avoid imputation bias, all features containing NaN or infinity values were removed. As a result, the ISCX-URL2016 phishing dataset was reduced to 69 remaining features.

The dataset contains both small and large numerical values, and unscaled data skews distance-based or gradient-based models. As mentioned earlier, the Min-Max scaling transforms all values to the [0, 1] range, ensuring that all features contribute equally to learning. Therefore, the Min-Max scaler is applied to all features in the ISCX-URL2016 phishing dataset [56], [57]. Table V shows a sample of the ISCX-URL2016 phishing dataset before and after normalization. Following normalization, feature selection improves model speed and generalization and reduces dimensionality by removing redundant or noisy features. The WOA and DA algorithms, along with the proposed union WOA and DA (WOAuDA) method, will be applied. Selected features are passed to classifiers for training. Overall, preprocessing ensures a clean, scaled, and optimized input space, which boosts performance and interpretability of the final model.

TABLE V. SAMPLE OF THE DATASET BEFORE AND AFTER NORMALIZATION

Before normalization	After normalization
80, 0.025, 5.0833335, 54, 48	0.138263666, 0.027443609, 0.154322353, 0.154121864, 0.157894737
78, 0.025641026, 3.1764705, 54, 40	0.131832797, 0.028147292, 0.096350222, 0.154121864, 0.131578947
71, 0.09859155, 6, 57, 22	0.109324759, 0.108228319, 0.182190694, 0.164874552, 0.072368421
64, 0.03125, 4.7, 53, 33	0.08681672, 0.034304511, 0.142668312, 0.150537634, 0.108552632
68, 0.5147059, 3.6923077, 48, 39	0.099678457, 0.565015502, 0.112032619, 0.132616487, 0.128289474
69, 0.028985508, 4.1666665, 50, 44	0.102893891, 0.031818678, 0.126453996, 0.139784946, 0.144736842
93, 0.021505376, 7.6, 72, 61	0.180064309, 0.023607405, 0.230833627, 0.218637993, 0.200657895
68, 0.029411765, 6.625, 54, 45	0.099678457, 0.032286599, 0.20119184, 0.154121864, 0.148026316
62, 0.032258064, 8.166667, 26, 17	0.080385852, 0.035411108, 0.248061343, 0.053763441, 0.055921053

B. Proposed WOAUDA Feature Selection Method

As mentioned earlier, feature selection reduces dimensionality by removing irrelevant or redundant features and improves model accuracy by eliminating noisy data. It enhances generalization by preventing overfitting and decreasing computational cost during model training and inference. In this work, we propose a union feature selection method that combines features selected by two selection techniques, namely DA and WOA metaheuristic algorithms, called WOAUDA. Union feature selection combines features selected by multiple selection techniques and merges different subsets to capture complementary information. It enhances

model robustness across diverse selection methods and reduces the risk of missing important but subtle features, while it also improves generalization by unifying stable feature patterns [34], [38], [40], [58], [59].

Union feature selection with metaheuristic algorithms combines feature subsets from multiple metaheuristic optimization algorithms and captures complementary patterns missed by individual search strategies. It reduces algorithm-specific bias in selected feature subsets and is useful when algorithms explore different regions of the search space. Additionally, it balances the exploration-exploitation strengths of different algorithms. Careful consideration must be given when selecting the two metaheuristic algorithms whose selected features will be combined. Combining features from

two metaheuristic algorithms may amplify redundancy and increase model complexity, as it can reinforce similar algorithmic bias and might not increase selection diversity meaningfully. Therefore, the best practices when uniting features from two metaheuristic algorithms involve selecting algorithms with complementary search behaviors and preferring algorithms with different convergence patterns. It is also important to validate each algorithm's stability and performance independently and to use harmonized fitness or selection criteria. The suitability of WOA and DA for Union Feature Selection lies in the fact that WOA and DA use different feature search strategies. Table VI presents the justification for selecting WOA and DA as complementary metaheuristic algorithms for union-based feature selection [34], [38], [40], [58], [59].

TABLE VI. RATIONALE FOR COMBINING WOA AND DA IN FEATURE SELECTION

#	Criteria	WOA	DA	Why It Matters
1	Complementary Search Dynamics	Exploits the best solution globally via spiral position updating. Balances exploration with adaptive coefficient a , decreasing linearly. Uses encircling behavior and spiral attack to converge fast.	Emphasizes local exploration using swarm behavior (separation, alignment, cohesion). Dynamically adapts between static (exploitation) and dynamic (exploration) swarm phases. Movement is influenced by social forces and neighborhood awareness.	Combining WOA's global convergence with DA's local swarm exploration creates diverse feature subsets, enhancing overall selection coverage.
2	Non-Overlapping Convergence Patterns	WOA converges toward a central best solution, refining it iteratively.	DA distributes search around multiple neighborhoods, guided by swarm dynamics.	WOA may focus on intensifying a narrow optimum, while DA explores peripheral and alternative regions, avoiding convergence overlap. Their union reduces selection bias.
3	Distinct Selection Tendencies	WOA often selects compact, high-impact feature subsets due to its tendency to exploit early-best solutions.	DA tends to retain a broader and more diverse feature set, reflecting the swarm's exploration of varied paths.	Their selected features are structurally and statistically different, leading to a union set that is richer and more informative than any single subset alone.
4	Fitness Function Harmonization	Both WOA and DA are compatible with multi-objective fitness functions. The function balances classification accuracy and feature compactness		It ensures that the union reflects features optimized under the same objective space.
5	Swarm Intelligence + Encircling Mechanism Synergy	DA's swarm coordination identifies local patterns (e.g., subdomains, arguments).	WOA's encircling mechanism captures global features (e.g., full URL length, entropy).	Combining these mechanisms improves coverage of local anomalies and global traits used in phishing attacks.

In our feature selection framework, we leverage the complementary strengths of two metaheuristic algorithms, WOA and DA, by combining their selected feature subsets using the union set theory function in mathematics. Formally, if S_{WOA} and S_{DA} denote the feature subsets selected by WOA and DA, respectively, the final feature subset S_{union} is obtained using Eq. (15). In other words, first, the WOA algorithm is executed to select a subset of relevant features. Subsequently, the DA algorithm is applied to obtain an additional subset of relevant features. The selected features from both algorithms are then combined into a unified feature set, ensuring that each feature appears only once. This final set of unique features is used for model training or evaluation. Algorithm 4 shows the pseudocode of the internal operations of the WOAUDA method.

$$S_{union} = S_{WOA} \cup S_{DA} = \{f_j | f_j \in S_{WOA} \text{ or } f_j \in S_{DA}\} \quad (15)$$

Algorithm 4: WOAUDA Pseudocode

```
Require: Best feature subset  $S_{WOA}$  from algorithm WOA,  
         best feature subset  $S_{DA}$  from algorithm DA  
Ensure: Final selected feature subset  $S_{union}$   
1:   Initialize  $S_{union} \leftarrow \emptyset$   
2:   for each feature  $f$  in  $S_{WOA}$  do  
3:     Add  $f$  to  $S_{union}$   
4:   end for  
5:   for each feature  $f$  in  $S_{DA}$  do  
6:     if  $f \notin S_{union}$  then  
7:       Add  $f$  to  $S_{union}$   
8:     end if  
9:   end for  
10:  Return  $S_{union}$ 
```

When applied to the ISCX-URL2016 phishing dataset, the WOA algorithm selected the following 43 features subset: {1, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 19, 20, 28, 29, 30, 33, 34, 35, 36, 37, 38, 40, 43, 45, 46, 47, 48, 49, 51, 52, 53, 55, 57, 58, 59, 62, 63, 69, 71, 72, 74, 75}. In parallel, the DA algorithm selected the following 46 features subset: {4, 7, 8, 11, 12, 13, 15, 16, 17, 18, 19, 20, 21, 22, 24, 27, 28, 29, 30, 31, 32, 33, 34, 36, 38, 39, 40, 41, 43, 44, 46, 47, 48, 52, 53, 54, 55, 56, 58, 62, 68, 69, 71, 72, 74, 75}. By applying the proposed WOAUDA union feature selection method to the outputs of both algorithms, the resulting combined 58 features subset was {1, 4, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 24, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 43, 44, 45, 46, 47, 48, 49, 51, 52, 53, 54, 55, 56, 57, 58, 59, 62, 63, 68, 69, 71, 72, 74, 75}. After removing duplicates, this unified set of unique features was used to train and evaluate the phishing detection model.

C. Classification

In order to build a strong phishing detector model, the SVM and XGBoost ML classifiers were adopted since they complement each other in challenging classification problems and have demonstrated successful use in detecting cybersecurity-related challenges. The broad ranges of hyperparameters were set as the initial values of each classifier, and the optimal values were chosen via tuning and evaluation. The RS algorithm was used to tune hyperparameters. It provides greater coverage of the search space and reduces evaluation. Domain knowledge and prior studies were used to define the tuning ranges so that they were relevant and effective.

SVM tuning was done using the kernel type, C, gamma, and the degree of the polynomials. An RBF kernel with $C = 10$ and gamma being auto-adjusted in the form of scale was used with the best configuration, whereas the degree of the polynomials was considered but not included in the final model. In the case of XGBoost, estimators, learning rate, maximum depth, minimum child weight, and gamma were tuned. The most suitable settings were 200 estimators, a learning rate of 0.05, a maximum depth of 7, a minimum child weight of 3, a subsample of 0.8, and gamma of 0.1.

The model was evaluated using five-fold cross-validation, where the dataset was divided into five approximately equal folds. In each iteration, four folds were used for training and one for validation. The reported performance metrics were calculated as the average across the five folds. The individual fold accuracies, along with their mean and standard deviation, are reported in Table VIII. The confusion matrices in Fig. 4–6 represent predictions obtained from one validation fold containing 3,072 samples. Hyperparameter configurations were selected based on their average validation performance across the five folds.

V. PROPOSED MODEL EVALUATION

This section assesses the phishing detection model proposed based on experiment analysis. It explains the implementation environment, states the assessment measures, and reports the key performance outcomes. To demonstrate the

effectiveness of the model, a comparison with existing methods is provided.

A. Implementation Environment

The experiments in this study were conducted on a high-end PC with very powerful specifications: An Intel Core i9-14900KS processor, an NVIDIA RTX 5090 GPU, 128 GB DDR5-6400, and Windows 11 Pro as the operating system. For implementation, this study uses Python 3.12 with scikit-learn and PyTorch/TensorFlow libraries.

The WOA and DA algorithms were configured with population sizes of 30 and maximum iteration counts of 40. A threshold of 0.5 was applied to convert continuous solutions into binary feature subsets. The fitness function was defined as $F(S) = (1 - \text{Accuracy}(S)) + 0.01(|S|/N)$, where S represents the selected feature subset, $|S|$ is the number of selected features, and N is the total number of available features. The function minimizes classification error while penalizing larger feature subsets to balance detection accuracy and feature reduction.

B. Confusion Matrix

The confusion matrix is used to analyze phishing detection with the help of sub-datasets chosen by the DA and WOA algorithms. It presents the results of classification and points out the wrong classifications, which gives an idea of the errors made in the model [60], [61], [62], [63]. Fig. 4(a) and 4(b) show the confusion matrices of XGBoost and SVM based on the proposed WOAUDA method, whereas Fig. 5 (a)-(b) and 6(a)-(b) show the results of WOA and DA, respectively. The extracted values are used to calculate Accuracy, Precision, Recall, and F1-score in order to compare the performance.

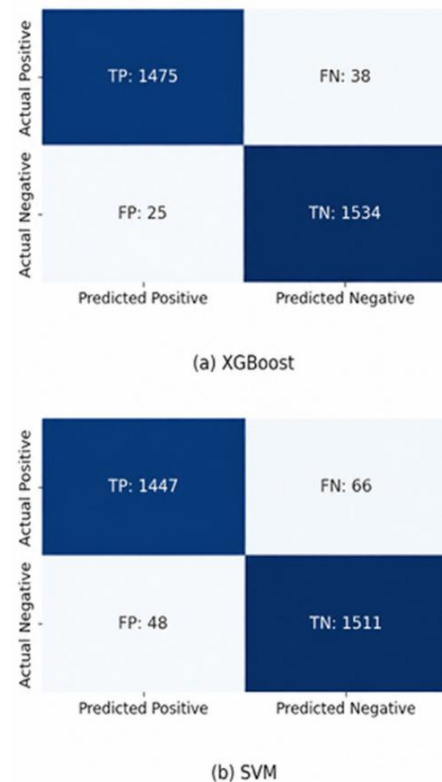
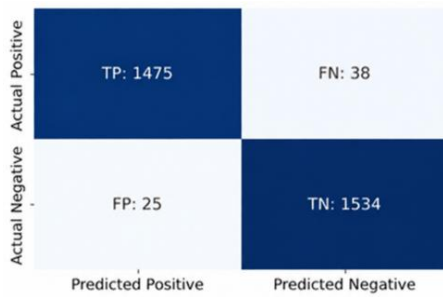


Fig. 4. WOAUDA confusion matrices.

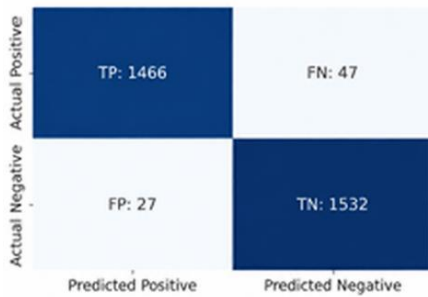


(a) XGBoost

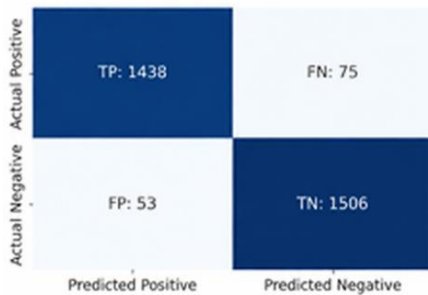


(b) SVM

Fig. 5. WOA confusion matrices.



(a) XGBoost



(b) SVM

Fig. 6. DA confusion matrices.

C. Experimental Results

Several metrics can be calculated using the values of the confusion matrix, each of which gives a different view of the performance of a classification issue, such as phishing attack

detection. One of the primary metrics is Accuracy, calculated using Eq. (16). Accuracy is used to view the performance of phishing classification across all classes. Another critical metric is recall, computed using Eq. (17). Recall measures how many actual spam emails are correctly identified. In cases where the cost of missing positive instances is high, recall is essential. One more prominent metric is precision, calculated using Eq. (18). Precision measures how many of the data points predicted as phishing are actually phishing. In cases where the cost of false positive instances is high, precision is important. Lastly, the F1-score is the harmonic mean of precision and recall, balancing both false positives and false negatives. It is calculated using Eq. (19) and is particularly useful when both types of errors carry significant costs [60], [61], [62], [63].

$$Accuracy = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (16)$$

$$Recall = \frac{TP}{(TP+FN)} \quad (17)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (18)$$

$$F1 - score = 2 \times \frac{Pre \times Rec}{Pre + Rec} \quad (19)$$

Besides, the ROC (Receiver Operating Characteristic) curve will be used to evaluate the performance of the proposed model at various threshold settings. The ROC curve compares the True Positive Rate (TPR) and the False Positive Rate (FPR). Eq. (20) and (21) are used to calculate the TPR and FPR, respectively. Each point on the ROC curve represents a TPR/FPR pair corresponding to a specific threshold. The Area Under the Curve (AUC) provides a single score to summarize the area under the ROC curve. AUC closer to 1 indicates better performance.

$$TPR = \frac{TP}{TP+FN} \quad (20)$$

$$FPR = \frac{FP}{FP+TN} \quad (21)$$

1) *Performance evaluation of the proposed phishing detection model:* This subsection outlines the performance of the proposed phishing detection model on the XGBoost and SVM classifiers based on the WOAUDA, WOA, and DA feature selection methods. The proposed phishing detection model will be evaluated using the Accuracy, Precision, Recall, and F1-score evaluation metrics. These measures can give recommendations on the effectiveness of the proposed phishing detection model on several aspects.

Fig. 7 shows the accuracy of the model. XGBoost achieves 97.95% accuracy with WOAUDA, compared with 97.92% using WOA and 97.59% using DA. These results represent improvements of 0.03 and 0.36 percentage points, respectively. Similarly, SVM achieves 96.29% accuracy with WOAUDA, representing an improvement of 0.46 percentage points over DA. Although WOAUDA achieves the highest recorded accuracy among the evaluated methods, these differences are relatively small and should be interpreted as observed performance improvements rather than evidence of statistical superiority.

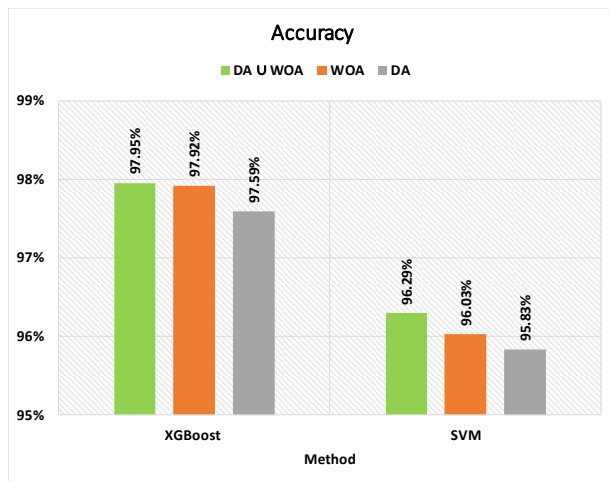


Fig. 7. Phishing classification accuracy.

Table VII presents the accuracy differences between the evaluated feature selection methods. WOA $\cup$ DA achieves improvements over DA of 0.46 and 0.36 percentage points for SVM and XGBoost, respectively. The corresponding improvements over WOA are 0.26 and 0.03 percentage points. These differences indicate modest performance gains under the evaluated experimental configuration, without establishing their statistical significance.

TABLE VII. DIFFERENCE IN ACCURACY (%) BETWEEN METHODS

Classifier	WOAUDA-WOA	WOAUDA-DA	WOA-DA
XGBoost	+0.03	+0.36	+0.33
SVM	+0.26	+0.46	+0.20

To further assess performance consistency, Table VIII presents the accuracy results across the five cross-validation folds for XGBoost and SVM using WOA $\cup$ DA, WOA, and DA. The average accuracy and standard deviation (SD) are also reported to evaluate performance variation across the folds.

TABLE VIII. FIVE-FOLD CROSS-VALIDATION ACCURACY (%)

Fold	XGBoost			SVM		
	WOAUDA	WOA	DA	WOAUDA	WOA	DA
1	97.90	97.96	97.51	96.22	95.97	95.77
2	97.99	97.85	97.63	96.34	96.08	95.89
3	97.94	97.93	97.57	96.27	96.01	95.81
4	98.03	98	97.68	96.39	96.12	95.86
5	97.89	97.86	97.56	96.23	95.97	95.82
Average	97.95	97.92	97.59	96.29	96.03	95.83
SD	0.060	0.064	0.066	0.073	0.067	0.046

The results show that WOA $\cup$ DA achieved the highest average accuracy for both classifiers, reaching 97.95% for XGBoost and 96.29% for SVM. The low standard deviations indicate limited accuracy variation across the five folds, suggesting consistent classification performance under the evaluated cross-validation setting.

Although accuracy provides a general view of the classifier performance, the Recall, Precision, and F1-score measures give a better understanding of the performance, as shown in Fig. 8-10. These measures can be used to measure the capacity of the classifier to detect positive samples correctly and to prevent false positives. Fig. 8 presents the recall results, showing that WOA $\cup$ DA achieves strong performance in identifying true positives. XGBoost achieved a recall of 97.49%, while SVM achieved 95.64% with WOA $\cup$ DA, compared with 95.04% using WOA and 94.84% using DA. This shows that WOAUDA is very effective in eliminating false negatives, which is a very important consideration in the cases of security detection and medical diagnostics. Fig. 9 shows that the Precision is high in all classifiers and methods with slight variations. WOAUDA attains the same accuracy as WOA on XGBoost (98.33%) and is also competitive with the SVM classifier. Fig. 10 shows that WOAUDA is robust in achieving balanced performance, which is confirmed by the F1-score.

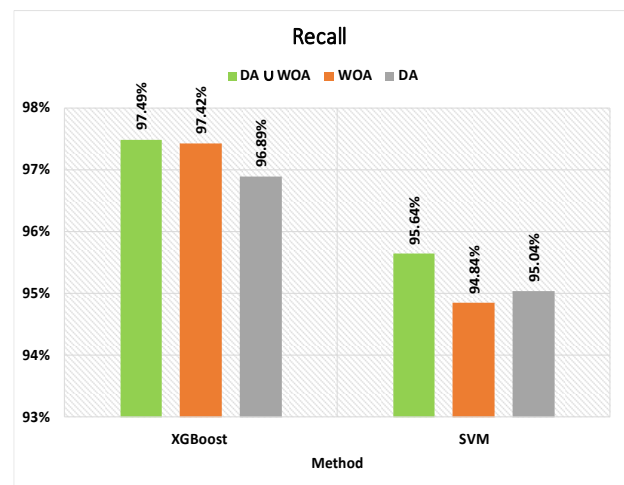


Fig. 8. Phishing classification recall.

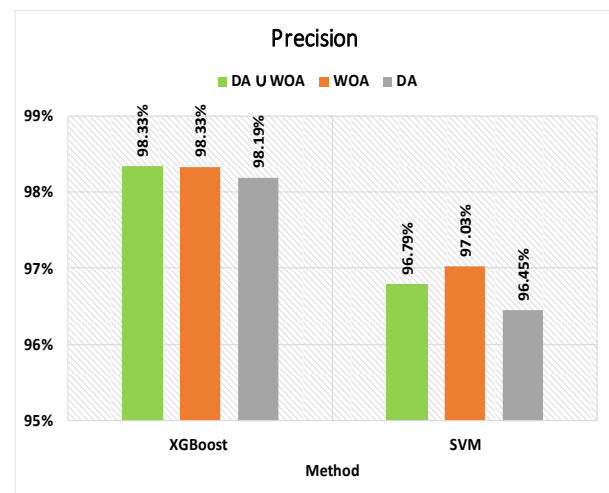


Fig. 9. Phishing classification Precision.

The F1-score represents the harmonic mean of precision and recall. It reflects classifier performance with respect to both false positives and false negatives. WOAUDA consistently achieves the highest F1-scores when used with

XGBoost and SVM classifiers. In particular, SVM performance improves to 96.21% with DA and to 96.21% with WOA (yielding F1-scores of 95.74% and 95.92%, respectively). These findings show that WOAUDA gives the most balanced classification performance.

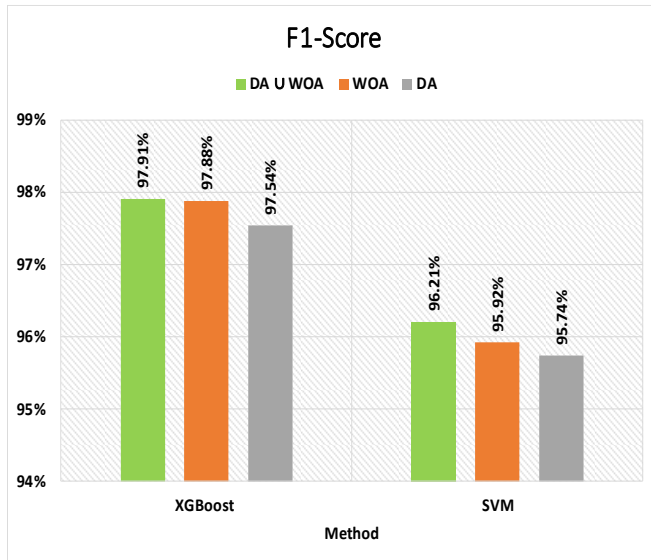


Fig. 10. Phishing classification F1-Score.

Overall, the experimental results show that $WOA \cup DA$ achieves competitive performance across Accuracy, Recall, Precision, and F1-score with both XGBoost and SVM classifiers. Although the proposed method achieves favorable results compared with the individual feature selection methods, the observed improvements require further statistical validation to establish their consistency and significance.

2) *Comparison with existing approaches:* Fig. 11 presents the accuracy comparison with existing methods evaluated on the ISCX-URL2016 dataset. The proposed XGBoost - $WOA \cup DA$ model achieved 97.95% accuracy, compared with 97.45% for [50] DNN, 94.73% for [50] Vanilla, 95.09% for [50] Sparse, 95.71% for [50] Denoising, and 93.18% for [50] CNN. It also achieved higher accuracy than [53] CNN-LSTM (96.85%). However, differences in evaluation protocols should be considered when interpreting these comparisons. Under the same experimental setting, the proposed method outperformed the XGBoost-WOA and XGBoost-DA variants by 0.03 and 0.36 percentage points, respectively. Fig. 12 presents an indicative accuracy comparison with other existing approaches evaluated on different datasets and under different experimental protocols. Although the proposed XGBoost - $WOA \cup DA$ model achieved competitive accuracy, these results cannot be directly compared due to differences in dataset characteristics, classification tasks, and evaluation settings. Therefore, the cross-dataset comparisons are presented for general performance context rather than as evidence of direct performance superiority.

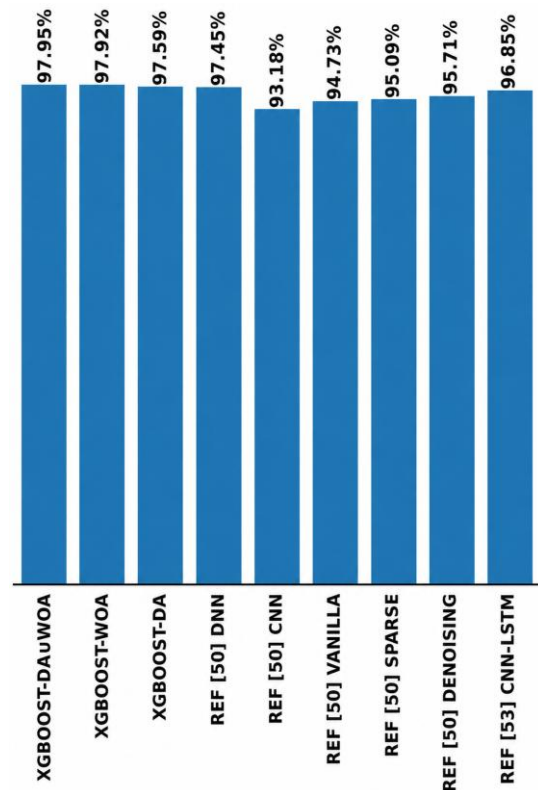


Fig. 11. Accuracy of proposed vs. existing models on ISCX-URL2016.

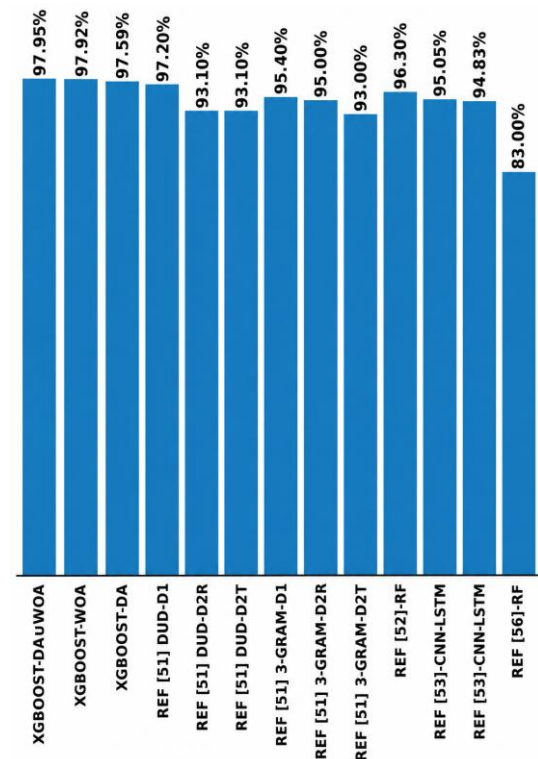


Fig. 12. Indicative accuracy comparison across different datasets and evaluation protocols.

VI. CONCLUSION

This study proposed a phishing detection framework centered on a novel union-based feature selection strategy that integrates the Whale Optimization Algorithm (WOA) and Dragonfly Algorithm (DA). Unlike single metaheuristic approaches, the proposed WOAUDA method combines the feature subsets selected by both algorithms to exploit their complementary search behaviors, enhancing feature diversity and reducing selection bias. Evaluated on the ISCX-URL2016 dataset using XGBoost and SVM classifiers with Random Search tuning and 5-fold cross-validation, the proposed method achieved superior performance. Specifically, XGBoost-WOAUDA attained 97.95% accuracy, outperforming XGBoost-WOA (97.92%) and XGBoost-DA (97.59%), while SVM improved to 96.29%, exceeding its DA-based counterpart by 0.46%. The model also demonstrated competitive Recall, Precision, and F1-score values. Comparative analysis with existing machine learning and deep learning models further indicated the effectiveness of the proposed approach under the evaluated experimental setting. However, repeated experimental runs and statistical significance tests are required to establish the consistency of these improvements.

REFERENCES

- [1] A. I. Karajeh, T. A. Al-Momani, M. A. Almomani, O. Almomani, and M. H. Tarawneh, "Forecasting future earnings via e-business information: Financial implications for investment decisions in the era of big data," *Investment Management and Financial Innovations*, vol. 22, no. 4, pp. 237–246, 2025, doi: 10.21511/imfi.22(4).2025.19.
- [2] A. A. S. Mohammad, H. A. Al-Ababneh, S. A. S. Alrhaimi, K. I. Al-Daoud, S. I. S. Mohammad, A. Vasudevan, A. W. P. Ling, S. T. Hai, and M. S. C. Rusuli, "Cloud and big data management: Integration into marketing strategies," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645, Springer, 2026, pp. 75–86, doi: 10.1007/978-3-032-10016-0_7.
- [3] Q. Y. Shambour, M. M. Abualhaj, and A. A. Abu-Shareha, "An effective doctor recommendation algorithm for online healthcare platforms," *Romanian Journal of Information Science and Technology*, vol. 27, no. 1, pp. 81–93, 2024, doi: 10.59277/ROMJIST.2024.1.06.
- [4] A. A. S. Mohammad, H. A. Al-Ababneh, K. Ibrahim, S. Mohammad, A. H. Almajali, A. Vasudevan, A. I. Mohammad, M. S. C. Rusuli, and V. A. Maniam, "Financial technologies for sustainable agriculture in developing countries," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645, Springer, 2026, pp. 27–38, doi: 10.1007/978-3-032-10016-0_3.
- [5] A. Vasudevan, S. Mohammad, A. Shlash, K. Al-Daoud, A. Mohammad, D. I. Mohammad, A. W. P. Ling, S. T. Hai, and M. S. C. Rusuli, "Trends and patterns in e-learning research: A bibliometric study on university student engagement," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645, Springer, 2026, pp. 203–214, doi: 10.1007/978-3-032-10016-0_17.
- [6] A. Vasudevan, S. Mohammad, and A. I. Mohammad, "Healthcare applications of artificial intelligence," in *Artificial Intelligence, Sustainable Technologies, and Business Innovation: Opportunities and Challenges of Digital Transformation*, A. Al-Sartawi and H. Ghura, Eds. Cham, Switzerland: Springer Nature, 2025, pp. 289–300.
- [7] A. A. S. Mohammad, S. I. S. Mohammad, A. Vasudevan, K. I. Al-Daoud, S. D. Ramachandaran, N. Alshdaifat, M. Alzyoud, N. Al-Shanableh, and M. S. C. Rusuli, "Teaching cybersecurity: Protecting the business environment," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645, Cham, Switzerland: Springer Nature, 2026, pp. 51–62, doi: 10.1007/978-3-032-10016-0_5.
- [8] M. N. Ghouri, G. Ahmed, A. S. Al-Shamayleh, S. Siddiqui, M. Maaz, and A. Akhunzada, "Next-gen IoT security: Deep learning-based detection of RPL attacks in mobile converged networks," *IEEE Open Journal of the Communications Society*, vol. 7, pp. 902–924, 2026, doi: 10.1109/OJCOMS.2026.3650763.
- [9] A. Ali, M. A. Akram, W. Farooq, M. Ali, M. Nazir, A. Muhammad, and T. Mazhar, "MalwareVison: A deep learning-driven approach for malware classification," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, 2025.
- [10] S. I. Mohammad, H. Y. Alsha'ar, R. M. I. Almajali, A. Al-Sarayreh, A. A. Y. Alzyoud, A. Vasudevan, B. Al Oraini, A. M. Al-Momani, and A. I. Mohammad, "Transforming industries with data: A bibliometric perspective on business analytics adoption," in *Artificial Intelligence, Sustainable Technologies, and Business Innovation: Opportunities and Challenges of Digital Transformation*, ser. Studies in Computational Intelligence, vol. 1171. Cham, Switzerland: Springer Nature, 2025, pp. 289–300, doi: 10.1007/978-3-031-77925-1_25.
- [11] A. A. E. M. Mohamed and I. Ismail, "A performance comparative on most popular internet web browsers," *Procedia Computer Science*, vol. 215, pp. 589–597, 2022, doi: 10.1016/j.procs.2022.12.061.
- [12] A. Chizhov and A. Fesenko, "Web hosting companies' client solutions: A study of a strategic standpoint," *Corporate & Business Strategy Review*, vol. 6, no. 1, Special Issue, pp. 421–429, 2025, doi: 10.22495/cbsrv6i1siart18.
- [13] T. Zhang, D. Jia, X. Fu, Z. Zhang, and Q. Liu, "WebAA: Website association analysis via multi-resource similarity computation," in *Computational Science – ICCS 2025*, ser. Lecture Notes in Computer Science, vol. 15904. Cham, Switzerland: Springer, 2025, pp. 50–64, doi: 10.1007/978-3-031-97629-2_4.
- [14] I. Qaddara, Y. Alraba'nah, and M. O. Hiari, "Evaluation of SQL and NoSQL databases on parallel processing," *Engineering, Technology & Applied Science Research*, vol. 15, no. 4, pp. 24298–24304, 2025.
- [15] A. Arif, M. Z. H. Ali, S. Z. Q. Haider, and Q. Niaz, "AI-driven edge computing for IoT: Revolutionizing phishing detection and mitigation," *Journal of Computing & Biomedical Informatics*, vol. 9, no. 1, 2025.
- [16] H. A. Anabousi, M. M. Abualhaj, A. Abu-Shareha, M. Yousif, M. Sh. Daoud, M. R. Faheem, and H. Soltani, "Malware detection using dynamic analysis with a modified Harris Hawk optimizer," *Discover Artificial Intelligence*, vol. 6, no. 1, pp. 1–30, 2026, doi: 10.1007/s44163-026-01359-0.
- [17] H. G. Alsheyab, M. M. Abualhaj, M. Al Zyoud, M. Yousif, and M. Sh. Daoud, "Web phishing detection using a modified spider monkey optimization algorithm," *Int. J. Advance Soft Comput. Appl.*, vol. 18, no. 3, pp. 133–152, 2026, doi: 10.15849/ijasca.125.
- [18] A. Saman and S. Rasool, "A feature-level hybrid model approach for automated phishing email detection," *Journal of Computing & Biomedical Informatics*, vol. 9, no. 1, 2025.
- [19] M. U. Javeed, S. M. Aslam, H. A. Sadiqa, A. Raza, M. M. Iqbal, and M. Akram, "Phishing website URL detection using a hybrid machine learning approach," *Journal of Computing & Biomedical Informatics*, vol. 9, no. 1, 2025.
- [20] N. Marshall, D. Sturman, and J. C. Auton, "Exploring the evidence for email phishing training: A scoping review," *Computers & Security*, vol. 139, Art. no. 103695, 2024, doi: 10.1016/j.cose.2023.103695.
- [21] B. Naqvi, K. Perova, A. Farooq, I. Makhdoom, S. Oyediji, and J. Porras, "Mitigation strategies against the phishing attacks: A systematic literature review," *Computers & Security*, vol. 132, Art. no. 103387, 2023, doi: 10.1016/j.cose.2023.103387.
- [22] B. Krebs, "PayPal phishing scam uses invoices sent via PayPal," *KrebsOnSecurity*, Aug. 18, 2022. [Online]. Available: <https://krebsonsecurity.com/2022/08/paypal-phishing-scam-uses-invoices-sent-via-paypal/>
- [23] D. Ali, M. M. Abualhaj, A. Abu-Shareha, M. Yousif, M. Sh. Daoud, M. R. Faheem, and H. Soltani, "Securing IoT networks against cyber threats using an improved whale optimization algorithm," *Journal of Artificial Intelligence and Technology*, 2026, doi: 10.37965/jait.2026.1033.

- [24] A. Shlash, S. I. S. Mohammad, H. A. Abu Owida, N. F. F. Alshdaifat, A. Vasudevan, A. I. Mohammad, H. N. Krishnasamy, A. Vasumathi, and P. R. Chiru, "Assessing machine learning approaches for real-time cardiovascular disease diagnosis," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645. Cham, Switzerland: Springer Nature, 2026, pp. 39–49, doi: 10.1007/978-3-032-10016-0_4.
- [25] M. N. Alqudah, N. Alshdaifat, H. A. Owida, S. Ibrahim, A. Vasudevan, C. Qian, A. I. Mohammad, R. D. Prasad, and W. Chen, "Building trust through interpretability in machine learning models for leukemia diagnosis," in *Artificial Intelligence and Technology: Systems Management, Decisions and Control for Sustainability in the Digital Age*, ser. Studies in Systems, Decision and Control, vol. 645. Cham, Switzerland: Springer Nature, 2026, pp. 127–137, doi: 10.1007/978-3-032-10016-0_11.
- [26] M. Yousef, A. M. Abu-Samaha, and S. Khawaj, "GA-SVM hybrid approach for optimizing feature selection in chronic kidney disease diagnosis," in *Proc. 2025 1st Int. Conf. Computational Intelligence Approaches and Applications (ICCIAA)*, Amman, Jordan, 2025, doi: 10.1109/ICCIAA65327.2025.11013723.
- [27] Q. Shambour, M. Al-Zyoud, and O. Almomani, "Quantum-inspired hybrid metaheuristic feature selection with SHAP for optimized and explainable spam detection," *Symmetry*, vol. 17, no. 10, Art. no. 1716, 2025, doi: 10.3390/sym17101716.
- [28] M. M. Abualhaj, S. N. Al-Khatib, A. Shalaldeh, M. Al-Zyoud, M. Sh. Daoud, H. Al-Mimi, and M. Anbar, "Intelligent malware detection through bio-inspired optimization and gradient boosting," *Journal of Advances in Information Technology*, vol. 17, no. 2, pp. 1–13, 2026.
- [29] R. Masadeh, O. Almomani, A. Zaqebah, S. Masadeh, K. Alshqurat, A. Sharieh, and N. Alsharman, "Narwhal optimizer: A nature-inspired optimization algorithm for solving complex optimization problems," *Computers, Materials & Continua*, vol. 85, no. 2, pp. 3709–3737, 2025, doi: 10.32604/cmc.2025.06679.
- [30] A. Al-Ou'n, S. A. Alomari, R. A. Zitar, M. Ismoilov, A. Smerat, W. Airbow, Z. Montazeri, M. Dehghani, O. P. Malik, and K. Eguchi, "Numbat optimization algorithm (NOA): A bio-inspired metaheuristic for solving optimization problems," *International Journal of Intelligent Engineering and Systems*, vol. 19, no. 2, pp. 996–1015, 2026, doi: 10.22266/ijies2026.0228.60.
- [31] H. Migdady, R. Abu Zitar, A. Smerat, Z. Montazeri, M. Dehghani, O. P. Malik, and K. Eguchi, "Tuatara optimization algorithm (TOA): A novel bio-inspired metaheuristic for constrained engineering design optimization," *International Journal of Intelligent Engineering and Systems*, vol. 19, no. 2, pp. 421–435, 2026, doi: 10.22266/ijies2026.0228.26.
- [32] R. A. Zitar, H. Migdady, M. Ismoilov, A. Smerat, W. Aribowo, Z. Montazeri, M. Dehghani, O. P. Malik, and K. Eguchi, "Solenodon optimization algorithm (SOA): A bio-inspired metaheuristic for efficient global optimization," *International Journal of Intelligent Engineering and Systems*, vol. 19, no. 2, pp. 636–652, 2026, doi: 10.22266/ijies2026.0228.40.
- [33] M. M. Abualhaj, H. Al-Mimi, M. Al-Zyoud, S. N. Al-Khatib, M. Sh. Daoud, H. Al-Aqrabi, M. Anbar, and A. Shalaldeh, "A bio inspired hybrid optimization framework for efficient real time malware detection," *Scientific Reports*, vol. 16, no. 1, Art. no. 4542, 2026, doi: 10.1038/s41598-025-33439-z.
- [34] M. M. Abualhaj, S. N. Al-Khatib, M. Al-Zyoud, I. Qaddara, M. O. Hiari, and S. M. A. Aldossary, "Enhanced network communication security through hybrid dragonfly-bat feature selection for intrusion detection," *Journal of Communications*, vol. 20, no. 5, pp. 607–618, 2025, doi: 10.12720/jcm.20.5.607-618.
- [35] O. Almomani, A. Alsaaidah, A. A. Abu-Shareha, A. Alzaqebah, M. A. Almaiah, and Q. Shambour, "Enhance URL defacement attack detection using particle swarm optimization and machine learning," *Journal of Computational and Cognitive Engineering*, vol. 4, no. 3, pp. 296–308, 2025, doi: 10.47852/bonviewJCCCE52024668.
- [36] Q. Abu Al-Haija and M. Al-Fayoumi, "An intelligent identification and classification system for malicious uniform resource locators (URLs)," *Neural Computing and Applications*, vol. 35, no. 23, pp. 16995–17011, 2023.
- [37] M. H. Nadimi-Shahraki, H. Zamani, Z. Asghari Varzaneh, and S. Mirjalili, "A systematic review of the whale optimization algorithm: Theoretical foundation, improvements, and hybridizations," *Archives of Computational Methods in Engineering*, vol. 30, no. 7, pp. 4113–4159, 2023.
- [38] S. Mirjalili, "Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053–1073, 2016.
- [39] M. Amin, K. M. O. Nahar, H. Gharaibeh, R. E. Al Mamlook, A. Nasayreh, N. Atitallah, A. Gharaibeh, R. Hamad, R. Abu Zitar, A. Smerat, and L. Abualigah, "A hybrid brain stroke prediction framework: Integrating feature selection, classification, and hyperparameter optimization," *Engineering Reports*, vol. 7, no. 7, Art. no. e70213, 2025, doi: 10.1002/eng2.70213.
- [40] M. Shokouhifar, F. Fanian, M. Hosseinzadeh, A. Smerat, K. M. Othman, A. Noorwali, and E. Y. O. Zafar, "Grey wolf optimizer for cluster-based routing in wireless sensor networks: A methodological survey," *Computer Modeling in Engineering & Sciences*, vol. 146, no. 1, Art. no. 6, 2026, doi: 10.32604/cmes.2026.073789.
- [41] R. Alabdallat, M. Abualhaj, and A. Abu-Shareha, "Android malware detection using a modified dwarf mongoose algorithm," *International Journal of Intelligent Engineering and Systems*, vol. 18, no. 8, Art. no. ijies2025.0930.21, 2025, doi: 10.22266/ijies2025.0930.21.
- [42] A. A. Abu-Shareha, M. M. Abualhaj, A. H. Hussein, A. Amer, A. Achuthan, and A. A. Halin, "Diabetes prediction using hybrid supervised and unsupervised techniques based on PIMA dataset," *Journal of Artificial Intelligence and Technology*, vol. 6, pp. 79–87, 2026, doi: 10.37965/jait.2025.0899.
- [43] B. Xu, K. Jummaa, F. M. A. Altalbawy, R. Roopashree, S. K. Saraswat, S. Ray, P. Tomar, S. Naem, H. H. Hamza, A. Smerat, and A. Khalid, "Machine learning-assisted computation of water activity for ionic liquid-based aqueous ternary elements," *Desalination and Water Treatment*, vol. 324, Art. no. 101484, 2025, doi: 10.1016/j.dwt.2025.101484.
- [44] M. Zubair, M. Owais, T. Hassan, et al., "An interpretable framework for gastric cancer classification using multi-channel attention mechanisms and transfer learning approach on histopathology images," *Scientific Reports*, vol. 15, Art. no. 13087, 2025, doi: 10.1038/s41598-025-97256-0.
- [45] Y. Alraba'nah, S. Al-Sharaeh, and G. Al Hindi, "Enhancing intrusion detection using hybrid long short-term memory and XGBoost," *Journal of Soft Computing and Data Mining*, vol. 6, no. 1, pp. 247–261, 2025, doi: 10.30880/jscdm.2025.06.01.016.
- [46] M. Ali, M. F. Mushtaq, U. Akram, S. Ramzan, S. Tahir, and M. Ahsan, "An ensemble approach for firewall log classification using stacked machine learning models," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, 2025.
- [47] A. Al Mosuli, M. Abualhaj, A. Abu-Shareha, M. Yousif, and M. Daoud, "Enhancing intrusion detection system performance using a modified grey wolf optimizer," *Annals of Emerging Technologies in Computing (AETiC)*, vol. 10, no. 1, pp. 21–44, Jan. 2026, doi: 10.33166/AETiC.2026.01.002.
- [48] B. Bischl, M. Binder, M. Lang, T. Pielok, J. Richter, S. Coors, et al., "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 13, no. 2, Art. no. e1484, 2023.
- [49] A.-C. Florea and R. Andonie, "Weighted random search for hyperparameter optimization," *International Journal of Computers Communications & Control*, vol. 14, no. 2, pp. 154–169, 2019, doi: 10.15837/ijccc.2019.2.3514.
- [50] M. K. Prabakaran, P. Meenakshi Sundaram, and A. D. Chandrasekar, "An enhanced deep learning-based phishing detection mechanism to effectively identify malicious URLs using variational autoencoders," *IET Information Security*, vol. 17, no. 3, pp. 423–440, 2023.

- [51] R. Vinayakumar, S. Srinivasan, K. P. Soman, and M. Alazab, "Malicious URL detection using deep learning," TechRxiv, preprint, 2023, doi: 10.36227/techrxiv.19308183.v1.
- [52] E. Gandotra and D. Gupta, "An efficient approach for phishing detection using machine learning," in *Multimedia Security, ser. Algorithms for Intelligent Systems*. Singapore: Springer, 2021, pp. 239–253, doi: 10.1007/978-981-15-8711-5_12.
- [53] S. J. Bu and H. J. Kim, "Optimized URL feature selection based on genetic-algorithm-embedded deep learning for phishing website detection," *Electronics*, vol. 11, no. 7, Art. no. 1090, 2022.
- [54] B. Geyik, K. Erensoy, and E. Kocyigit, "Detection of phishing websites from URLs by using classification techniques on WEKA," in *Proc. 2021 6th Int. Conf. Inventive Computation Technologies (ICICT)*, Coimbatore, India, 2021, pp. 120–125, doi: 10.1109/ICICT50816.2021.9358642.
- [55] T. B. Tariq, S. N. Khosa, M. Z. Hadi, T. Kiran, M. Mansab, U. Akram, and M. F. Mushtaq, "Intelligent cyber security framework for threat detection using ensemble learning techniques," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, 2025, doi: 10.56979/802/2025.
- [56] M. M. Abualhaj, S. N. Al-Khatib, M. O. Hiari, Q. Y. Shambour, A. Al-Allawee, O. Almomani, M. Sh. Daoud, and M. Anbar, "An efficient feature selection technique to enhance spam email detection," *TEM Journal*, vol. 15, no. 1, pp. 91–101, 2026, doi: 10.18421/TEM151-09.
- [57] M. Sarfraz, I. A. Sumra, B. Khalid, and E. Fatima, "AI-driven predictive threat detection and cyber risk mitigation: A survey," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, pp. 1–18, 2025, doi: 10.56979/802/2025.
- [58] U. Humayun, M. Irfan, S. Bibi, K. Bibi, and H. Shokat, "Jellyfish algorithm for feature selection: Improving machine learning-based heart disease prediction," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, 2025, doi: 10.56979/802/2025.
- [59] M. Sarfraz, I. A. Sumra, B. Khalid, and E. Fatima, "AI-driven predictive threat detection and cyber risk mitigation: A survey," *Journal of Computing & Biomedical Informatics*, vol. 8, no. 2, Mar. 2025, doi: 10.56979/802/2025.
- [60] N. M. Alaskar, M. Hussain, S. J. Almheiri, Atta-ur-Rahman, A. Khan, and K. M. Adnan, "Big data-driven federated learning model for scalable and privacy-preserving cyber threat detection in IoT-enabled healthcare systems," *Computers, Materials & Continua*, vol. 87, no. 1, Art. no. 29, 2026, doi: 10.32604/cmc.2025.074041.
- [61] M. Hassan, M. Zubair, R. Ibrahim, T. Mahmood, M. Hussain and M. Umair, "Explainable and Trustworthy AI in Oncology: A Systematic Review Across Major Organ Systems," in *IEEE Access*, vol. 14, pp. 127800-127833, 2026, doi: 10.1109/ACCESS.2026.3717851.
- [62] Q. Y. Shambour, A. A. Abu-Shareha, and M. M. Abualhaj, "A hotel recommender system based on multi-criteria collaborative filtering," *Information Technology and Control*, vol. 51, no. 2, pp. 390–402, 2022, doi: 10.5755/j01.itc.51.2.30701.
- [63] M. D. Majid, M. Anwar, S. F. Bilal, M. Hussain, M. Zubair, J. Sultana, and M. A. Habib, "A hybrid learning framework for automated multiclass electrocardiogram classification with SimCardioNet," *Scientific Reports*, vol. 16, no. 1, Art. no. 7621, 2026, doi: 10.1038/s41598-026-36932-1