Received: 28 October 2024 | Revised: 16 December 2024 | Accepted: 1 January 2025 | Published online: 14 February 2025

RESEARCH ARTICLE

Journal of Computational and Cognitive Engineering
2025, Vol. 4(3) 296–308
DOI: 10.47852/bonviewJCCE52024668

BON VIEW PUBLISHING

# Enhance URL Defacement Attack Detection Using Particle Swarm Optimization and Machine Learning

**Omar Almomani[1,*], Adeeb Alsaaidah[1], Ahmad Adel Abu-Shareha[2], Abdullah Alzaqebah[3], Mohammed Amin Almaiah[4] and Qusai Shambour[5]**

[1]*Department of Networks and Cybersecurity, Al-Ahliyya Amman University, Jordan*

[2]*Department of Data Science and Artificial Intelligence, Al-Ahliyya Amman University, Jordan*

[3]*Department of Computer Science, The World Islamic Sciences and Education University, Jordan*

[4]*Department of Computer Science, The University of Jordan, Jordan*

[5]*Department of Software Engineering, Al-Ahliyya Amman University, Jordan*

**Abstract:** Uniform resource locator (URL) defacement attack can be defined as any cyberattack in which the attacker replaces the appearance or content of the targeted webpage with their own that is intended to disgrace, mislead, or malign the website. Detecting URL defacement attacks is significant to avoid breaching the security of the website content or its configuration files, modifying the file locations, templates, or attacks on the website environment and applications. A machine learning (ML) technique can be used to detect the defacement attack on any website with complex content and structure, as opposed to the classical techniques for detection, such as Diff comparison, Document Object model tree analysis, and checksum, which can only be applied to static websites. This article proposes a feature selection model based on particle swarm optimization with support vector machine, decision tree, random forest, Naive Bayes, and k-nearest neighbor ML classification algorithms. The proposed model aims to improve the URL defacement attack detection by selecting the best features from the ISCX-URL-2016 dataset. Then, the reduced set of features produced by the proposed model step is used as input to evaluate and compare the results of the used ML classifiers. The results showed that the proposed model has significantly reduced the features, regarding the classification's feature reduction, the random forest classifier outperformed other classifiers in terms of true positive rates, accuracy, precision, sensitivity, and F-measure, whereas the proposed model with random forest classifier has 99.21% True positive rates, 99.29% accuracy rate, 99.38% precision rate, 99.21% sensitivity rate, and 99.29% F-measure rate. In the future directions of this article, more research should be done on a variety of things, including varying and sophisticated techniques of altering the URL defacement since it would better calibrate the model for application in real-life situations.

**Keywords:** URL defacement, machine learning, particle swarm optimization (PSO), cybersecurity, ISCX-URL-2016 dataset

## 1. Introduction

The rapid expansion of web applications and online services led to the importance of uniform resource locator (URL) as a key asset for all organizations, businesses, and even governments. URL is defined as the address used to access resources on the internet. URL malware, URL phishing, URL spam, and URL defacement are examples of URL cyberthreats. URL defacement attack is a penetration of website files, content, or appearances. Unlike other cyberattacks like. URL malware, URL phishing, and URL spam, URL defacement attacks involve no phishing, robbing, or stealing crimes or uncovering confidential information. The defacement attack aims to breach the security of the website files or its configuration, modify the files' locations or templates, or attack the website environment and applications. Defacement attacks commonly replace the website's content with the hacker's message. Such URL defacement attacks embarrass website owners and ruin their reputations, especially for government parties, companies, international associations, and organization [1]. The defacement commonly reveals political disagreement and conflicts, or it promotes hackers in the cybersecurity community. There are various other purposes for the attacker to implement the URL defacement attack, including revenge, asking for money to restore the original website contents, or proving the attacker's ability to attack secure websites. Accordingly, defacement attack detection in an early stage is required [2].

A defacement attack is a way to modify a website's content, structure, or configuration using various techniques, including SQL injection, malware infection, unauthorized access, and DNS

***Corresponding author:** Omar Almomani, Department of Networks and Cybersecurity, Al-Ahliyya Amman University, Jordan. Email: o.almomani@ammanu.edu.jo

hijacking, making it hard to defend. The consequences of such attacks are obvious for both the owners and users of the victim websites. An example of such an attack is the DNS hijacking of the Google website in 2012, in which an Algerian hacker named MCA-CRB conducted a defacement attack and led to the falsification of the Romanian on their server instead of the Google web page for more than an hour. MCA-CRB also performed an attack on almost 5,530 websites all over the world. In 2018, the patient survey website hosted by the UK National Health Service (NHS) was hacked by AnoaGhost, which was the message that appeared on the website for a few hours due to that attack. In 2019 a hacker breached the hosting provider's system, called Pro-Service in Georgia, and defaced 15,000 government, bank, and press websites. In 2020, more than 51 US government websites were defaced by two Iranian hackers in response to the assassination of the military general Qasem Soleimani in Iraq. The hackers display messages and show images against US President Donald Trump. Various defacement attacks have been conducted worldwide [3].

The classical techniques for defending against defacement attacks include scanning security vulnerabilities, web attack monitoring, and defacement monitoring tools [4]. The scanning tools aim at discovering the vulnerabilities of the websites and the hosts to be fixed. These tools focus on the operating systems and the device settings. Various tools have been developed, such as Abbey Scan and Acunetix Vulnerability Scanner. The web attack monitoring tools analyze the traffic and discover any suspicious patterns. The defacement attack monitoring tools are used as comparison techniques to detect the differences between the original website content and structure and the target website version. Site24x7 and WebOrion are two examples of defacement monitor tools. The problem with the defacement attack monitoring tools is their inability to be used with dynamic websites [5].

Detecting URL defacements and other attacks, such as DoS, phishing, sniffing, ping sweeps, port scanning, and malware, can be implemented using machine learning (ML) methods and algorithms [6]. The process involves capturing the network flow, extracting prominent features, analyzing the content, and identifying potential threats. Accordingly, URL defacement attack detection can be implemented using supervised or unsupervised ML. The unsupervised algorithms identified patterns but required further mapping between patterns and the final class label, as the attack may involve multiple patterns, and so for the normal traffic [7]. The supervised algorithms produced the final output and more accurate results than the unsupervised algorithms [8]. Accordingly, a classification algorithm is trained to detect the attack along with the important step of feature selection, which has the advantage of simplifying the training and detection phases by minimizing features, which reduces the time and space required for the classification processes and improves the results of the classification process [9]. Therefore, in this article, supervised algorithm is used in the classification stage.

Feature selection is implemented either using a filter-based or wrapper-based approach [10]. The filter approach selects a feature-by-feature to reduce the complexity of testing all possible combinations of features equal to n! where n is represented by the feature number. The filter-based approach selects the features based on their correlation with the output class. The wrapper-based approach scores selected subsets of features based on their performance with a specific classification algorithm. Accordingly, the wrapper-based contributions to the accuracy of the classification output. Yet, testing all feature subsets is computationally expensive, which can be reduced using optimization algorithms, such as particle swarm optimization (PSO) [11].

Overall, various algorithms for feature selection were proposed based on bio-inspired algorithms [12, 13], and various supervised ML algorithms were used for classification. Yet, integrating feature selection algorithms and ML algorithms is critical to obtain adequate results for the URL defacement attack detection task. Accordingly, this article used PSO for feature selection with various ML classification algorithms, including naive Bayes (NB), k-nearest neighbors (KNN), support vector machine (SVM), decision tree J48 (DT), and random forest (RF).

The contributions to the research are summarized as follows:

1) The study provides a model for improving the URL defacement by decreasing the number of selected features of ISCX-URL-2016 dataset using (PSO). The model reduces features to 38 out of 79.
2) The study evaluates the proposed model using (NB), (KNN), (SVM), (DT), and (RF) machine learning (ML) classifiers. RF classifier is superior to other tested classifiers in term of true-positive rates, accuracy, precision, sensitivity, and F-measure.

The article's remaining sections are arranged as follows: In Section 2, the relevant literature is reviewed, along with an overview of the defacement attack and the elements of the proposed solution. The methodology is detailed in Section 3. Sections 4 and 5 contain the findings and the conclusion, respectively.

## 2. Related Works and Background

This section offers background information and an overview of the subject being studied in this paper.

### 2.1. Machine learning and classification

ML is the study of how machines can learn from data. Unsupervised ML is a subfield of ML that concerns learning without guidance. Accordingly, given data samples represented by a feature vector X, unsupervised ML focuses on discovering relations and patterns between these samples. Supervised ML is concerned about learning with guidance. Classification and regression are among the common tasks in supervised learning. In the classification task, a training set of $n$ samples is represented by $(x_1, y_1)$, $(x_2, y_2)$,.., $(x_n, y_n)$, where $x_i$ is the feature vector for sample $i$, and $y_i$ is the output class label of that sample. These samples are used to train the classifier and build a trained model. As a result, the trained model is represented as a mapping function $X \rightarrow Y$, limited to the $Y$ values being made available to the classifier in the training phase [14].

The classification algorithms can be implemented using various classification algorithms, which can be categorized into instance-based, DT-based, probability-based, artificial neural network (ANN), and SVM [15]. Instance-based work does not use any trained models. Instead, this technique uses the instances of the training set as a reference to produce the correct class label for the samples of the unknown class. Accordingly, there is no time consumption for the training phase. Besides, it can be easily adapted to new instances. DT builds a trained model in the form of a tree. This technique's advantages are interoperability, accuracy, and low memory consumption. The probability-based technique computes the maximum likelihood of a class label based on a given input. Like the decision-tree technique, probability classifiers are interoperable and require low memory space. Yet, it is inefficient with complex data and commonly produces bad results for complex problems. An ANN builds a trained model in the form of a

connected network that is trained to optimize the weights of the edges between the nodes. This technique's advantage is the ability to handle complex and noisy data. The disadvantages are the time required for the training process and the possibility of generating bad results for complex problems. An SVM builds a model by drawing a hyperplane between two data classes. Commonly, SVM produces good results for complex nonlinear data. Overall, no best classifier can be used for all data types, and different classifiers produce different results for different inputs [16, 17].

Accordingly, more than a single classifier is commonly utilized, and the results are evaluated and compared. This article uses five classifiers: KNN, SVM, DT, NB, and RF. KNN [18] is an instance-based classification algorithm that does not build any models. Instead, the mapping function of the KNN can be represented as finding the value of $y$ as most of the values of $y$' for samples with feature vectors x' are most similar to $x$. SVM builds a trained model in the form of a hyperplane model iteratively. The hyperplane between two classes is a line that divides the space into $x_1$ and $x_2$ and divides the output class labels into $y_1$ and $y_2$ with minimal error and maximum margin. DT builds a trained model in the form of a tree that can be represented as a tree with branches and nodes. The internal branches test a feature's value in the feature space, while the leaves represent a class label. The tree is constructed as the branches aim to maximize the separation between the classes based on their feature values. NB is a probability-based mechanism for classification and prediction based on input and historical data. RF builds a model of multiple decision trees with randomly selected features. The goal is to create a set of uncorrelated trees, and then use them to make a collective decision [17].

## 2.2. Particle swarm optimization

PSO, invented by Kennedy and Eberhart in 1995 [19], and modified version of the PSO was introduced in 1998 [20]. PSO is a population-based algorithm inspired by swarms' social interactions and flocking behavior, such as fish and birds. PSO is implemented by randomly initiating candidate solutions (particles) that evolved to create a new population from the old one iteratively until reaching the best solution. Each candidate solution in a given population is moved (i.e., changed) based on the current velocity and distance to the local and global best solution. Figure 1 shows the PSO algorithm flowchart diagram, the local best solution is the best fitness value obtained by that solution, and the global best solution is the best fitness achieved by any solution. Various optimization techniques have been proposed to explore the solution space of complex problems and find the optimal solution defined by a specific objective function. Compared to other existing meta-heuristic optimization techniques, such as GA, Tabu Search, ACO, Grey Wolf Optimization, and Harris Hawks Optimization, PSO has various advantages, which can be summarized as follows:

1) PSO makes a few assumptions about the problem being solved. Thus, it depends on simple problem representation and a few parameter settings.
2) PSO is simple and can be used to search for a large solution space.
3) PSO does not require the problem to be of differential form.
4) PSO can solve complex problems [21].

## 2.3. Related work

ML techniques are frequently used to identify malicious websites, network attacks, and cybercrimes, and these tasks may be reduced to a classification issue. To train an ML for a malicious website detection system, a classifier is used with the available data that are relevant to the classification of authentic and malicious websites. Previous studies proved that when robust ML classifiers are utilized, malicious website accuracy of detection can be high. Yet, some preprocessing is required to obtain such results. Among these, feature reduction is highly important. A diversity of feature selection methods is utilized to reduce the feature number.

Massive malicious website detection methods have been presented using various classifiers and preprocessing steps. James et al. [22] proposed a method that involved analyzing URLs and using the following ML classifiers: NB, DT, KNN, and SVM. The proposed method examined the hostname and path and categorized it as an attack or a genuine website. The strategy for detecting phishing websites depends on lexical characteristics, host properties, and page significance. The evaluations were conducted using datasets from Alexa, Phishtank, and others. Similarly, Subasi et al. [23] used the ANN, KNN, SVM, RF, rotation forest, and C4.5 for malicious website detection. The results asserted that the RF could only be accurate up to 97.26%. Other classifiers all achieved the same accuracy as that reported in the study.

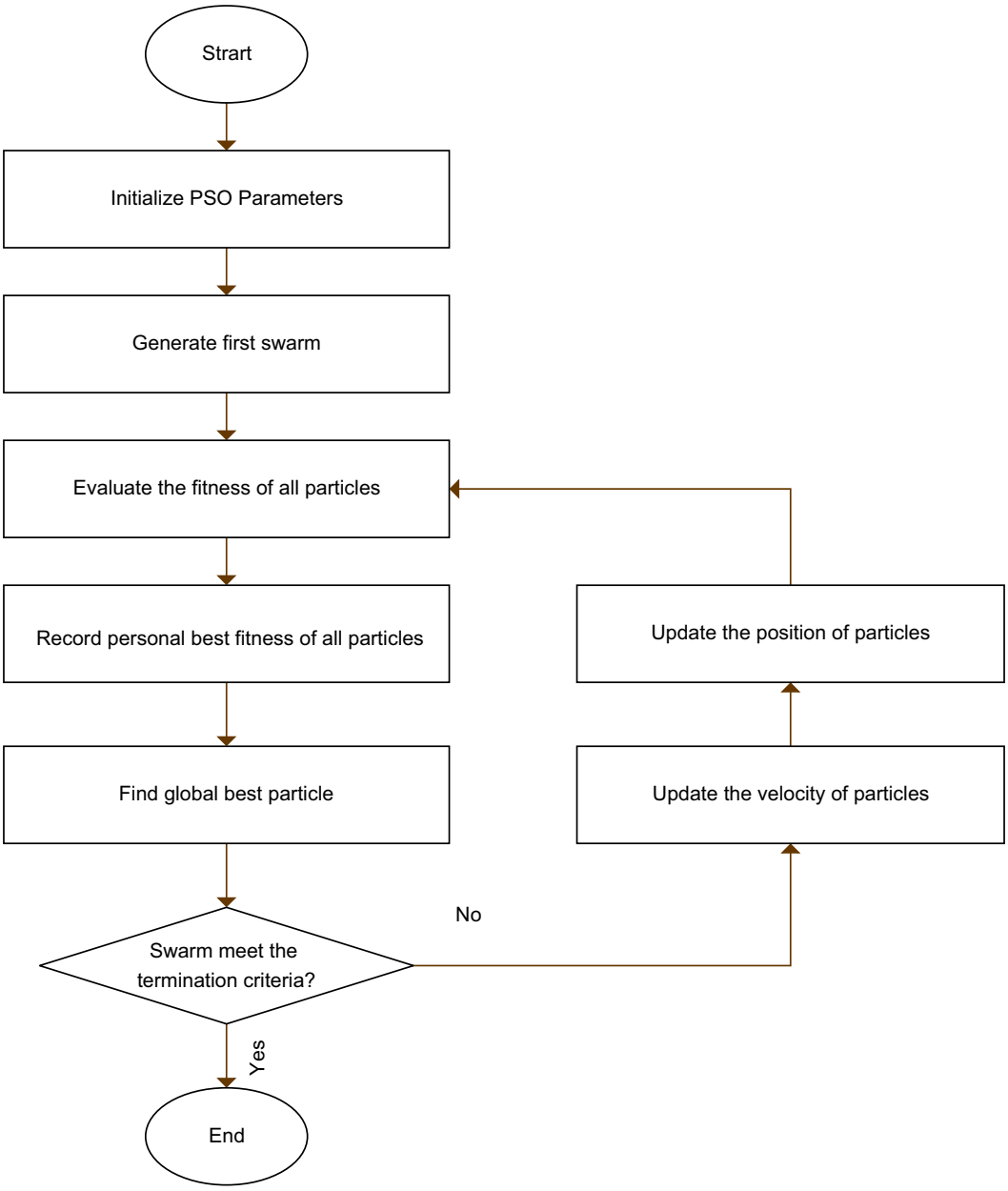
Hota et al. [24] proposed a feature selection technique known as the remove-replace feature selection technique (RRFST). RRFST removes features from the original feature space if removing that feature maintained or improved the accuracy of the results; otherwise, the feature is replaced in the original feature space. Two decision rree (DT) techniques, C4.5 and classification and regression tree (CART), as classifiers, which were then combined to create an effective classification model with a smaller feature subset produced through RRFST. The experimental results showed that an accuracy of 99.27% was achieved utilizing C4.5 and CART ensemble with just 11 features.

Jain and Gupta [25] proposed an anti-phishing technique based on ML with features retrieved from the client side. The experiments have been performed on multiple datasets. First, the Phishtank dataset was used with 1528 phishing websites, followed by the Openphish dataset with 613 phishing websites, the Alexa dataset with 1600 legitimate websites, the Payment Gateway dataset with 66 samples, and the Top Banking website dataset with 252 legitimate websites. The proposed technique used a feature extraction method on the client's side and improved the phishing detection accuracy by using ML techniques such as RF, SVM, NN, LR, and NB neural networks. Joshi and Pattanshetti [26] used the relief-F algorithm as a feature selection approach and the RF method as a binary classifier. The experiments were conducted based on datasets obtained from the Mendeley website. Ubing et al. [27] proposed a malicious website detection technique using ensemble learning. To implement ensemble learning, three strategies were used: bagging, boosting, and stacking. The experiments were conducted based on a dataset of 30 features and 5126 records from UCI, an open public datasets website. The classifiers were combined to get the highest accuracy possible from a DT.

Mao et al. [28] proposed a mechanism for learning-based aggregate analysis to assess page layout similarity for phishing page detection. Four learning classifiers were used such as SVM, DT, AdaBoost, and RF. The experiments were conducted based on a dataset obtained from the Phishtank dataset. Sahingoz et al. [29] proposed a real-time anti-phishing system that used lexical features and seven different classification algorithms. The experiments were conducted based on a new dataset. The FR method, which only used lexical features, performed the best for phishing URL detection, with an accuracy rate of 97.98%.

Zamir et al. [30] proposed a stacking-based supervised learning model for phishing website detection. The goal was to increase the classification accuracy by stacking the most effective classifiers and

**Figure 1**
**PSO algorithm flowchart diagram**




using features selected using the PCA. To build two stacking models, they integrated random forest (RF), neural network (NN), and bagging algorithms in one, and k-nearest neighbors, random forest, and bagging algorithms in the other. As for the accuracy performance, which was 97.4%, the RF-NN-Bagging technique surpassed all other models mentioned in the research. Similarly, Subasi and Kremic [31] proposed an intelligent system for identifying phishing websites. A customized ML model was used to distinguish between legitimate and fraudulent websites. To create a reliable and clever phishing website-detecting structure, several classification algorithms were applied. Receiver operating characteristic (ROC), area under ROC curve (AUC), and F-measure were used to evaluate the effectiveness of ML techniques. Adaboost with SVM outperformed all other classification techniques, according to the results, reaching the maximum accuracy of 97.61%.

Ali and Malebary [32] proposed a technique that weighted the processed websites based on PSO. To improve the accuracy of phishing website detection, PSO is used to identify elements in the websites and weigh them based on their significance in distinguishing between legitimate and malicious websites. Results demonstrated that the proposed PSO-based component weighting increased the ML models' ability to distinguish between legitimate and phishing websites. Alsariera et al. [33] proposed the ABET, RoFET, BET, and LBET models. These models combine the extra tree classifier with a meta-learner model. A meta-algorithm, also known as a metaheuristic, is a high-level technique created to

identify the best answer to an optimization issue. This technique was used to construct M1, rotation forest, bagging, and LogitBoost. The UCI dataset was resampled in this study using 10-fold cross-validation, and the extra tree model was trained and tested ten times, with the results evaluated using a weighted average value. According to the experimental findings, the four fusion models achieved superior results, with an accuracy of 97% or more, false-negative rates under the value of 0.038, and false-positive rates under the value of 0.019.

Odeh et al. [34] proposed PhiBoost using adaptive boosting and feature selection for phishing website detection. The experiments were conducted based on datasets collected from the Phishtank archive, MillerSmiles archive, and Google search engines. The proposed approach generates a very high predicted accuracy of almost 99%. Harinahalli Lokesh and BoreGowda [35] proposed a phishing detection system using wrapper-based feature selection. Different ML methods were used as classifiers, including RF, kNN, DT, and SVM. The utilized dataset was obtained from UCI. In comparison to other techniques, RF was found to achieve the highest accuracy of 96.87%.

Gupta et al. [36] proposed a phishing detection approach based on an RF. The experiments were conducted based on the ISCX-URL-2016 dataset, which includes 19,964 instances with 9 lexical features. The ISCX-URL-2016 dataset has more than 35,300 legal URLs and over 10,000 phishing URLs. In the experiments, 10,000 innocuous URLs were randomly selected with 9964 phishing URLs. The performance of the RF was compared with four single classifiers, and the results showed that RF had the best accuracy of 99.57%. However, SVM performed better than RF in terms of response time. A summary of the related work is given in Table 1.

## 3. Methodology

A model for defacement attacks is proposed using feature selection and ML techniques. Figure 2 illustrates the steps of the proposed model. The proposed model uses the PSO algorithm to reduce features and the SVM, RF, NB, K-NN, and J48 ML classifiers to classify the defacement attacks.

### 3.1. Preprocessing stage

The preprocessing stage involves various straightforward steps, including missing values removal, eliminating duplicated data, data transformation, and data normalization. This stage aims to minimize overfitting, remove outliers, and ease feature selection and classification processes. As such, eliminating duplicated data improves classification outcomes by distinguishing infrequent records from frequent ones and eliminating bias toward such infrequent yet duplicated records. The records with missing values are removed to eliminate classifier confusion and incorrect results [37]. Numerical data were scaled using a min-max method to the range of [0,1], as given in Equation (1).

$$X_{Normalized} = \frac{\mathrm{X} - X_{Minimum}}{X_{Maximum} - X_{Minimum}} \tag{1}$$

**Table 1**
**Summary of the related work**

| Author | Classifiers | Feature selection | Datasets | Results | |
|---|---|---|---|---|---|
| James et al. [22] | RT, KNN, SVM, and NB | Lexical feature extraction | www.alexa.com, www.dmoz.org, personal web browser history, and www.phishtak.com | RT<br>KNN<br>SVM<br>NB | 85.63%<br>75.77%<br>74.48%<br>83.50% |
| Subasi et al. [23] | RF | No | UCI | 97.36% | |
| Hota et al. [24] | CART and C4.5 | RRFST | Khonji's Anti-Phishing | 99.27% | |
| Jain and Gupta [25] | RF, SVM, NN, LR, and NB | Client-side specific features | Phishtank, Openphish, Alexa, Payment Gateway, and Top Banking Website | RF<br>SVM<br>KNN<br>LR<br>NB | 99.09%<br>96.16%<br>98.05%<br>98.25%<br>97.59% |
| Joshi and Pattanshetti [26] | RF | ReliefF | Mendeley website | 97.63% | |
| Ubing et al. [27] | EL | RFG | UCI | 95.4% | |
| Mao et al. [28] | SVM, RF, DT, and AB | Learning-based aggregation | Phishtank | SVM<br>RF<br>DT<br>AB<br>DT<br>Adaboost | 96.9%<br>97.3%<br>93.6%<br>94.5%<br>97.02%<br>93.24% |

(*Continued*)

**Table 1**
**(*Continued*)**

| Author | Classifiers | Feature selection | Datasets | Results | |
|---|---|---|---|---|---|
| Sahingoz et al. [29] | DT, Adaboost, Kstar, KNN, RF, SMO, and NB | NLP | Ebbu2017 Phishing Dataset | Kstar | 93.65% |
| | | | | KNN | 95.67% |
| | | | | RF | 97.98% |
| | | | | SMO | 94.92% |
| | | | | NB | 95.67% |
| Zamir et al. [30] | Stacking (RF + NN + bagging) | IG, gain ratio, relief-F, and recursive feature | Phishing Websites | Stacking (RF + NN + bagging) | 97.4% |
| Subasi and Kremic [31] | Adaboost with SVM | no | UCI | Adaboost with SVM | 97.61% |
| Ali and Malebary [32] | BPNN, SVM, C4.5, NBRF and KNN | PSO-based feature weighting | UCI | BPNN | 96.43% |
| | | | | SVM | 92.19% |
| | | | | C4.5 | 96.28% |
| | | | | NB | 91.03% |
| | | | | RF | 96.83% |
| | | | | KNN | 96.32% |
| Alsariera et al. [33] | ABET, RoFET, BET, and LBET | Gain ratio, IG, RFE, PCA and relief-F | UCI | RoFET | 97.45% |
| | | | | BET | 97.4% |
| | | | | LBET | 97.58% |
| | | | | ABET | 97.49% |
| Odeh et al. [34] | PhiBoost | PhiBoost | PhishTank, MillerSmiles, and Google Searching Operators | 98.9% | |
| Harinahalli Lokesh and BoreGowda [35] | SVM, SVC, KNN, DT, and RF | Manually | UCI | SVM | 48.56% |
| | | | | SVC | 92.69% |
| | | | | KNN | 93.53% |
| | | | | DT | 96.05% |
| | | | | RF | 96.78% |
| Gupta et al. [36] | RF, KNN, SVM, and LR | New feature extraction mechanism | ISCXURL-2016 | RF | 99.57% |
| | | | | KNN | 99.04% |
| | | | | SVM | 97.64% |
| | | | | LR | 95.56% |

where the $X_{Normalized}$ parameter is the normalized value, while the $X_{Maximum}$ and $X_{Minimum}$ parameters represent the maximum and the minimum values in the data vector, respectively.

## 3.2. PSO feature selection

PSO is an optimization algorithm that can be used for complex problems, as discussed in Section 2.3. Feature selection can be represented as a binary optimization problem of 0s and 1s, where the value 0 or 1 is assigned to each feature. As a wrapper-based feature selection approach, using PSO for feature selection, at each iteration, the solution evolved with a different combination of 0s and 1s. Each solution is evaluated and evolved to obtain the optimal solution.

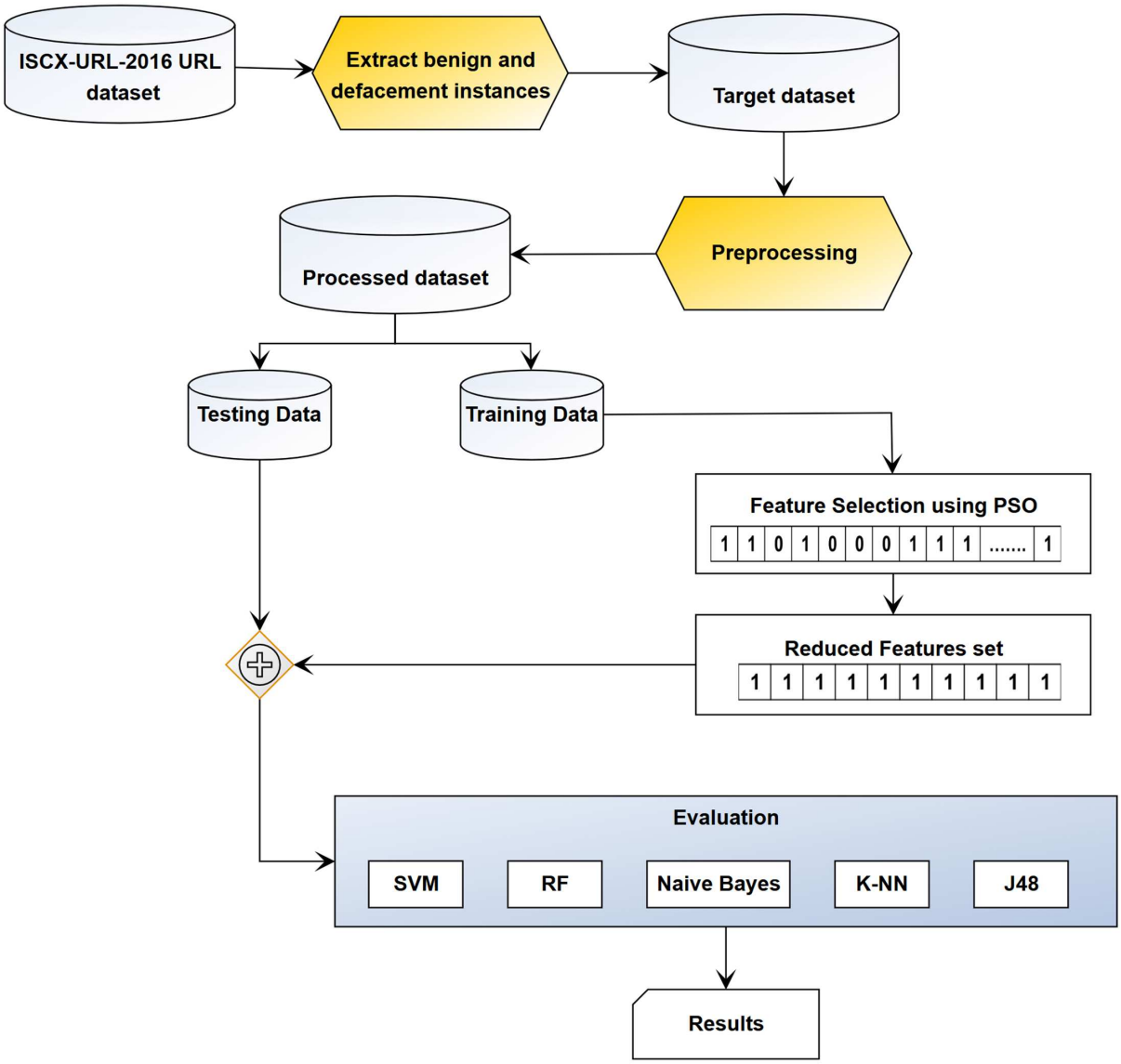
In the process, binary solutions are initialized randomly, each for each particle (i.e., search agent), and the fitness of each solution is calculated accordingly. Then, PSO selects the individual solution (pbest), and the global solution (gbest) is based on fitness values. Next, the positions and velocities of the particles will be updated to evolve toward the optimal solution, which means generating new solutions. PSO updates the velocity and position according to Equations (2) and (3), respectively. Iteratively, PSO calculates the fitness value for each new solution, compares the fitness values with the current best solution, and updates the best solution according to the fitness value. The output of the PSO-based feature selection will represent the optimal or near-optimal solution, which is the most informative and relevant set of features. Algorithm 1 shows the pseudocode for solving feature selection using the PSO.

$$v_{id}^{k+1} = v_{id}^{k} + c_1 r_1^k \left(pbest_{id}^{k} - x_{id}^{k}\right) + c_2 r_2^k \left(gbest_{id}^{k} - x_{id}^{k}\right) \quad (2)$$

$$x_{id}^{k} = x_{id}^{k} + v_{id}^{k+1} \quad (3)$$

where $v_{id}^{k}$ and $x_{id}^{k}$ are the particle's ith speed at its k occurrences and position in the d dimension. $pbest_{id}^{k}$ is the individual's best solution, while $gbest_{id}^{k}$ illustrates the best global solution. The variables c1 and c2 are control parameters, while r1 and r2 are random values in the range of [0,1].

**Figure 2**
**The model methodology**




**Algorithm 1: Pseudocode for the PSO for feature selection**

**Step 1**: Initialization: initialize the positions and velocity for each particle randomly, positions [0,1]
**Step 2**: While the stop condition is not met, do
for each particle x ∈ {1,2,...,N} loop

1) Calculate the fitness of $x_1$
2) If $x_1$ fitness < pbest
   - pbest = $x_1$
   - pbestfit = fitness of $x_1$
3) If pbest < gbest
   - gbest = pbest
   - gbestfit = pbestfit

**Step 3**: Update the particle's velocity and position according to Equation (2) and Equation (3)
**Step 4**: Return gbest and gbestfit

### 3.3. Machine learning classifiers

As mentioned earlier, various classifiers are used in the proposed model as different classifiers produce different results [16, 17]. SVM is used for its well-known accurate outputs for various problems. SVM is a binary classifier that uses a hyperplane to separate the data into two classes. The hyperplane maximizes the margin's space between the classes depending on the utilized kernel. Because it is a component of the structural risk minimization strategy, the SVM offers high generalization capabilities [38, 39].

J48 classifier is a tree-based classifier with an enhanced method for pruning trees to cut down on classification-related issues. First, J48 creates a decision tree by choosing the root attribute with the highest gain value, and then the branches are constructed for the values of the attributes. Once all branches have the same class output, the process is repeated for each branch [40].

RF is an ensemble learning classifier of multiple weak trees based on merging the predictions of different trees, each of which is trained independently. RF can overcome overfitting, shorten the

training time, and work well with massive datasets. RF can be implemented with the missing values of an incomplete dataset [41].

NB classifier is an extension of the conventional technique, the maximum likelihood estimation theory (MLE). NB uses a probability-based mechanism for the classification and prediction of the class of a sample based on input attributes and historical data. Moreover, Naïve Bayes rules rely on the fundamentals of the conditional probability rule. However, it enforces independence between the attributes [42].

KNN uses majority voting to classify a sample based on its closest K-NN neighborhoods. The distance metric used to identify the closest neighbors, such as the Euclidean distance, significantly impacts its performance. The K-NN classifier calculates the differences between the sample to be classified and the neighborhood. The value of K affects the performance [43].

## 4. Experiments and Results

The experiments were conducted using a Windows 11 operating system, an i7-1065G7 processor running at 1.50 GHz, and 16.0 GB of RAM. The experiments are conducted using Python programming language. For the right comparison, we apply the same conditions and variables, such as population size and the maximum number of iterations for all the compared methods and techniques. Using trial and error, the population size of the PSO was set to 30, and the maximum number of iterations was set to 100 in the experiments. For each experiment, 50 runs are performed.

### 4.1. Dataset

The ISCX-URL-2016 URL dataset is used for experiments, a publicly available dataset from the Canadian Institute for Cybersecurity. Five URL malicious classes, including benign, malware, defacement, spam, and phishing, are included in the dataset. The dataset contains 79 lexical features from URLs, as listed in Table 2 [44].

The ISCX-URL-2016 dataset includes 12,000 spam samples, 10,000 phishing samples, 11,500 malware samples, 45,500 defacement samples, and more than 35,000 benign samples. The benign samples are the legitimate URLs that do not direct users to malicious websites or attempt to install harmful malware on their computers are known as benign URLs. Although these websites

**Table 2**
**Features of dataset**

| No | Feature | No | Feature |
|---|---|---|---|
| **F1** | Querylength | F41 | Directory_DigitCount |
| **F2** | domain_token_count | F42 | File_name_DigitCount |
| **F3** | path_token_count | F43 | Extension_DigitCount |
| **F3** | avgdomaintokenlen | F44 | Query_DigitCount |
| **F5** | longdomaintokenlen | F45 | URL_Letter_Count |
| **F6** | avgpathtokenlen | F46 | host_letter_count |
| **F7** | tld | F47 | Directory_LetterCount |
| **F8** | charcompvowels | F48 | Filename_LetterCount |
| **F9** | charcompace | F49 | Extension_LetterCount |
| **F10** | ldl_url | F50 | Query_LetterCount |
| **F11** | ldl_domain | F51 | LongestPathTokenLength |
| **F12** | ldl_path | F52 | Domain_LongestWordLength |
| **F13** | ldl_filename | F53 | Path_LongestWordLength |
| **F14** | ldl_getArg | F54 | sub-Directory_LongestWordLength |
| **F15** | dld_url | F55 | Arguments_LongestWordLength |
| **F16** | dld_domain | F56 | URL_sensitiveWord |
| **F17** | dld_path | F57 | URLQueries_variable |
| **F18** | dld_filename | F58 | spcharUrl |
| **F19** | dld_getArg | F59 | delimeter_Domain |
| **F20** | urlLen | F60 | delimeter_path |
| **F21** | domainlength | F61 | delimeter_Count |
| **F22** | pathLength | F62 | NumberRate_URL |
| **F23** | subDirLen | F63 | NumberRate_Domain |
| **F24** | fileNameLen | F64 | NumberRate_DirectoryName |
| **F25** | this.fileExtLen | F65 | NumberRate_FileName |
| **F26** | ArgLen | F66 | NumberRate_Extension |

(*Continued*)

**Table 2**
**(*Continued*)**

| No | Feature | No | Feature |
|---|---|---|---|
| **F27** | pathurlRatio | F67 | NumberRate_AfterPath |
| **F28** | ArgUrlRatio | F68 | SymbolCount_URL |
| **F29** | argDomanRatio | F69 | SymbolCount_Domain |
| **F30** | domainUrlRatio | F70 | SymbolCount_Directoryname |
| **F31** | pathDomainRatio | F71 | SymbolCount_FileName |
| **F32** | argPathRatio | F72 | SymbolCount_Extension |
| **F33** | executable | F73 | SymbolCount_Afterpath |
| **F34** | isPortEighty | F74 | Entropy_URL |
| **F35** | NumberofDotsinURL | F75 | Entropy_Domain |
| **F36** | ISIpAddressInDomainName | F76 | Entropy_DirectoryName |
| **F37** | CharacterContinuityRate | F77 | Entropy_Filename |
| **F38** | LongestVariableValue | F78 | Entropy_Extension |
| **F39** | URL_DigitCount | F79 | Entropy_Afterpath |
| **F40** | host_DigitCount | Lable | benign, malware, defacement, spam, and phishing |

of legitimate URLs may include advertisements and adware, the adware is normally safe for a computer. Around 35,300 benign URLs were gathered from well-known Alexa websites. To retrieve the URLs of the websites, the domains were run via a Heritrix web crawler, which resulted in the crawling of around 500,000 distinct URLs. Then, duplicated and malicious URLs were removed from the initial list.

Malicious URLs direct users to a website that usually downloads harmful software, which can be used to steal identities, corrupt data, and even record keystrokes. Software that is harmful to computers and can steal personal data is known as malware. Malware is a threat with harmful and dangerous consequences, like the consequences of biological agents or terrorist cells looking to cause chaos. Malware can take many forms, including scareware, spyware, and ransomware. From the DNS-BH project, which maintains a list of malware sites, more than 11,500 URLs associated with malware websites were acquired.

Defacing a URL typically entails modifying some aspect of it, such as the way it looks or a part of the material on it. There are several reasons why hackers attempt to deface a website. This type of action is taken when certain website information needs to be updated without the consent of the website's owner, and strictly speaking, it entails breaking into a website. The defacement URL category includes more than 45,450 URLs. They are highly regarded websites by Alexa that host harmful website content with concealed or fake URLs.

Phishing URLs typically lure a visitor to a bogus website where they attempt to collect as much personal information as they can. Sometimes, a simple URL error might easily direct a user to a phishing website. Phishing uses social engineering tactics by hackers to acquire sensitive data, such as credit card numbers and other digital identities. Spam links send unsolicited emails to advertise or seriously harm the recipient's computer. From Open Phish, a database of live phishing websites, approximately 10,000 phishing URLs were obtained.

Spam URLs are regularly included in spam emails. Certain spam URLs might be dangerous and corrupt the user's PC with spyware and adware. From the publicly accessible WEBSPAM-UK2007 dataset, about 12,000 spam URLs were gathered.

This article focuses on detecting and classifying the defacement attack instead of others since this type has the most significant number of instances recorded in the dataset. For that, and as the first step in the proposed model, we extract only the defacement and benign samples to form the target dataset for which the model will be trained.

## 4.2. Evaluation metrics

True positive (TP), true negative (TN), false positive (FP), and false negative (FN) ratios are used to assess the effectiveness of the model, which depend on the number of correctly and incorrectly classified samples as presented in the confusion matrix in Table 3. Other metrics, including accuracy, precision, sensitivity, and F-Measure are calculated using these measurements, as given in Equations (4), (5), (6), and (7).

**Table 3**
**Confusion matrix**

| | | Predicted | |
|---|---|---|---|
| | | Normal | Attack |
| Actual | Normal | TP | FN |
| | Attack | FP | TN |

$$\text{Accuracy} = TP + TN/TP + TN + FP + FN \tag{4}$$

$$\text{Precision} = TP/TP + FP \tag{5}$$

$$\text{Sensitivity} = TP/TP + FN \tag{6}$$

$$\text{F} - \text{Measure} = 2 * \text{Precision} * \text{Sensitivity}/\text{Precision} + \text{Sensitivity} \tag{7}$$

## 4.3. Results and discussion

The feature selection step using PSO is combined with a KNN-based wrapper model in the experiments. Accuracy, precision, sensitivity, F-measure, TP, and the number of selected features is the six performance metrics that are used to evaluate the efficiency of the proposed PSO feature selection model to reach the optimal value. The optimized feature selection results are compared to the model's results using all the features, the best features reported in the literature, and Information Gain (IG). These results are evaluated with classification algorithms, including SVM, J48, RF, and NB.

*4.3.1. Features selection results*

The selected features using the proposed model are listed in Table 4. As noted, the PSO reduced the number of features to 38 features instead of 79 features for detecting defacement attacks. Accordingly, the proposed PSO optimizes the balance between removing unnecessary features and enhancing performance in terms of F-measure, accuracy, precision, TP, and sensitivity.

*4.3.2. Classification results*

The developed PSO model is assessed using the SVM, J48, RF, NB, and KNN ML classifiers. The results of the classification experiments are presented in Table 5.

The proposed model achieved significant results, as illustrated in Figure 3. Compared to the results of the other feature selection techniques and using all features, the proposed model based on PSO achieved the highest TP rate with a TP of 99.21%. The proposed model's best results were achieved using the random forest classifier. Similarly, the proposed model achieved better accuracy, precision, sensitivity, and F-measure than others. Using the RF classifier, the proposed model achieved an accuracy of 99.29%, a precision of 99.38%, a sensitivity of 99.21%, an F-measure of 99.29%. False-positive rates of 0.63%. The RF classifier was the best among other classifiers because it increases accuracy and

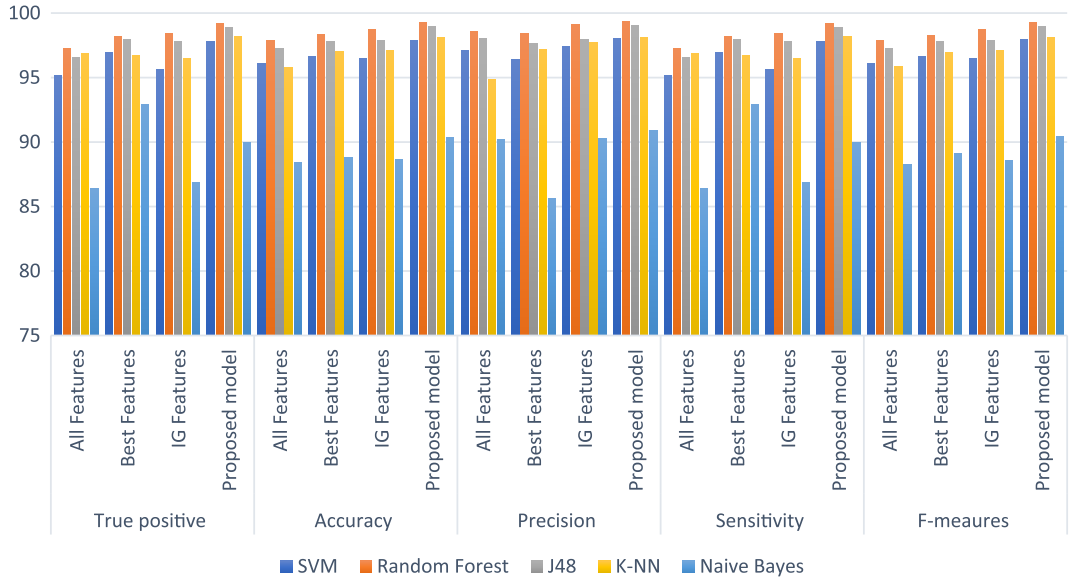
**Table 4**
**Selected feature set based on PSO**

| Item | Description |
|---|---|
| **Features** | F1,F3,F4,F8,F10,F14,F15,F17,F18,F23,F24,F25,F29,F31,F35,F36,F39,F40,F41,F43,F46,F48,F49,F50,F60,F61, F62,F63,F64,F65,F67,F68,F70,F72,F74,F76,F77,F78 |
| **Total** | 38 |

**Table 5**
**Results of SVM, J48, RF, NB, and KNN**

| Classifier | Feature Reduction | TP % | FN % | FP % | TN % | Accuracy % | Precision % | Sensitivity % | F-measure % |
|---|---|---|---|---|---|---|---|---|---|
| RF | All features | 97.24 | 2.76 | 1.43 | 98.57 | 97.90 | 98.58 | 97.24 | 97.91 |
| | Best features | 98.15 | 1.85 | 1.54 | 98.46 | 98.31 | 98.43 | 98.15 | 98.29 |
| | Information gain | 98.40 | 1.60 | 0.93 | 99.07 | 98.73 | 99.09 | 98.40 | 98.74 |
| | Proposed model | 99.21 | 0.79 | 0.63 | 99.37 | 99.29 | 99.38 | 99.21 | 99.29 |
| SVM | All features | 95.16 | 4.84 | 2.90 | 97.10 | 96.12 | 97.09 | 95.16 | 96.12 |
| | Best features | 96.91 | 3.09 | 3.62 | 96.38 | 96.65 | 96.37 | 96.91 | 96.64 |
| | Information gain | 95.65 | 4.35 | 2.62 | 97.38 | 96.51 | 97.38 | 95.65 | 96.51 |
| | Proposed model | 97.81 | 2.19 | 2.00 | 98.00 | 97.90 | 98.03 | 97.81 | 97.92 |
| KNN | All features | 96.85 | 3.15 | 5.32 | 94.68 | 95.77 | 94.89 | 96.85 | 95.86 |
| | Best features | 96.71 | 3.29 | 2.74 | 97.26 | 96.99 | 97.20 | 96.71 | 96.95 |
| | Information gain | 96.44 | 3.56 | 2.26 | 97.74 | 97.08 | 97.75 | 96.44 | 97.09 |
| | Proposed model | 98.18 | 1.82 | 1.94 | 98.06 | 98.12 | 98.10 | 98.18 | 98.14 |
| J48 | All features | 96.53 | 3.47 | 2.00 | 98.00 | 97.26 | 98.00 | 96.53 | 97.26 |
| | Best features | 97.97 | 2.03 | 2.33 | 97.67 | 97.82 | 97.63 | 97.97 | 97.80 |
| | Information gain | 97.82 | 2.18 | 2.06 | 97.94 | 97.88 | 97.98 | 97.82 | 97.90 |
| | Proposed model | 98.87 | 1.13 | 1.02 | 98.98 | 98.92 | 99.00 | 98.87 | 98.93 |
| NB | All features | 86.42 | 13.58 | 9.54 | 90.46 | 88.42 | 90.23 | 86.42 | 88.28 |
| | Best features | 92.88 | 7.12 | 15.27 | 84.73 | 88.77 | 85.65 | 92.88 | 89.12 |
| | Information gain | 86.87 | 13.13 | 9.50 | 90.50 | 88.67 | 90.31 | 86.87 | 88.56 |
| | Proposed model | 89.97 | 10.03 | 9.21 | 90.79 | 90.38 | 90.87 | 89.97 | 90.42 |

**Figure 3**
**Classification results**




decreases overfitting by combining the predictions from multiple decision trees.

## 5. Conclusions and Future Work

This article proposed a model for feature selection based on PSO feature selection for URL defacement attacks. Using fewer and more effective selected features leads to faster classification and better performance. The ISCX-URL-2016 dataset with SVM, J48, RF, NB, and KNN classifiers was used to assess the proposed model. Two stages made up the experiment's execution. The first stage focuses on feature selection using a PSO, while the second stage involves assessing the proposed model using SVM,

J48, RF, NB, and KNN ML classifiers. The results of the first stage demonstrated that the proposed model reduced the number of features to 38 out of 79. Accordingly, the results of the second stage demonstrated that the proposed model with an RF classifier produces the highest rate in terms of TP rates, accuracy, precision, sensitivity, and F-measure compared to other models. In future work, more metaheuristic algorithms, URL attacks, and deep learning approaches will be developed, implemented, and evaluated to boost performance.

## Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

## Data Availability Statement

The data that support the findings of this study are openly available in URL dataset (ISCX-URL2016) at https://www.unb.ca/cic/datasets/url-2016.html.

## Author Contribution Statement

**Omar Almomani:** Conceptualization, Methodology, Software, Investigation, Writing – original draft, Supervision, Project administration. **Adeeb Alsaaidah:** Conceptualization, Formal analysis, Writing – original draft. **Ahmad Adel Abu-Shareha:** Methodology, Formal analysis, Investigation, Data curation, Writing – review & editing, Visualization. **Abdullah Alzaqebah:** Software, Resources, Data curation, Writing – review & editing, Visualization. **Mohammed Amin Almaiah:** Validation, Resources, Writing – original draft. **Qusai Shambour:** Resources, Writing – original draft.

## References

[1] Burruss, G. W., Howell, C. J., Maimon, D., & Wang, F. (2022). Website defacer classification: A finite mixture model approach. *Social Science Computer Review, 40*(3), 775–787. https://doi.org/10.1177/0894439321994232

[2] Nguyen, T. H., Hoang, X. D., & Nguyen, D. D. (2021). Detecting website defacement attacks using web-page text and image features. *International Journal of Advanced Computer Science and Applications, 12*(7), 215–222. https://doi.org/10.14569/IJACSA.2021.0120725

[3] Hoang, X. D., & Nguyen T. H. (2021). A CNN-based model for detecting website defacements. *Journal of Science & Technology on Information and Communications, 1*(1), 4–9.

[4] Abualhaj, M. M., Al-Shamayleh, A. S., Munther, A., Alkhatib, S. N., Hiari, M. O., & Anbar, M. (2024). Enhancing spyware detection by utilizing decision trees with hyperparameter optimization. *Bulletin of Electrical Engineering Informatics, 13*(5), 3653–3662. https://doi.org/10.11591/eei.v13i5.7939

[5] Dau, H. X., Trang, N. T. T., & Hung, N. T. (2022). A survey of tools and techniques for web attack detection. *Journal of Science and Technology on Information Security, 1*(15), 109–118. https://doi.org/10.54654/isj.v1i15.852

[6] Banerjee, S., Swearingen, T., Shillair, R., Bauer, J. M., Holt, T., & Ross, A. (2022). Using machine learning to examine cyber-attack motivations on web defacement data. *Social Science Computer Review, 40*(4), 914–932. https://doi.org/10.1177/0894439321994234

[7] Jeyabharathi, Alphonse, A. S., Priya, E. L. D., & Kowsigan, M. (2022). Review of machine learning techniques used for intrusion and malware detection in WSNs and IoT devices. In S. L. Tripathi, D. K. Singh, S. Padmanaban, & P. Raja (Eds.), *Design and development of efficient energy systems* (pp. 57-65). Wiley. https://doi.org/10.1002/9781119761785.ch5

[8] Naeem, S., Ali, A., Anam, S., & Ahmed, M. M. (2023). An unsupervised machine learning algorithms: Comprehensive review. *International Journal of Computing Digital Systems, 13*(1), 911–921. http://dx.doi.org/10.12785/ijcds/130172

[9] Bergadano, F., Carretto, F., Cogno, F., & Ragno, D. (2019). Defacement detection with passive adversaries. *Algorithms, 12*(8), 150. https://doi.org/10.3390/a12080150

[10] Ali, M. Z., Abdullah, A., Zaki, A. M., Rizk, F. H., Eid, M. M., & El-Kenway, E. M. (2024). Advances and challenges in feature selection methods: A comprehensive review. *Journal of Artificial Intelligence and Metaheuristics, 7*(1), 67–77. https://doi.org/10.54216/JAIM.070105

[11] Agrawal, P., Abutarboush, H. F., Ganesh, T., & Mohamed, A. W. (2021). Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019). *IEEE Access, 9,* 26766–26791. https://doi.org/10.1109/ACCESS.2021.3056407

[12] Pham, T. H., & Raahemi, B. (2023). Bio-inspired feature selection algorithms with their applications: A systematic literature review. *IEEE Access, 11,* 43733–43758. https://doi.org/10.1109/ACCESS.2023.3272556

[13] Alsaaidah, A., Almomani, O., Abu-Shareha, A. A., Abualhaj, M. M., & Achuthan, A. (2024). ARP spoofing attack detection model in IoT network using machine learning: Complexity vs. accuracy. *Journal of Applied Data Sciences, 5*(4), 1850–1860. https://doi.org/10.47738/jads.v5i4.374

[14] Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets, 31*(3), 685–695. https://doi.org/10.1007/s12525-021-00475-2

[15] Almomani, O. (2021). A hybrid model using bio-inspired metaheuristic algorithms for network intrusion detection system. *Computers, Materials & Continua, 68*(1), 409–429. https://doi.org/10.32604/cmc.2021.016113

[16] Bashir, A. K., Khan, S., Prabadevi, B., Deepa, N., Alnumay, W. S., Gadekallu, T. R., & Maddikunta, P. K. R. (2021). Comparative analysis of machine learning algorithms for prediction of smart grid stability. *International Transactions on Electrical Energy Systems, 31*(9), e12706. https://doi.org/10.1002/2050-7038.12706

[17] Lim, K. S., Lee, L. H., & Sim, Y.-W. (2021). A review of machine learning algorithms for fraud detection in credit card transaction. *International Journal of Computer Science and Network Security, 21*(9), 31–40. https://doi.org/10.22937/IJCSNS.2021.21.9.4

[18] Abualhaj, M. M., Abu-Shareha, A. A., Shambour, Q. Y., Alsaaidah, A., Al-Khatib, S. N., & Anbar, M. (2024). Customized K-nearest neighbors' algorithm for malware detection. *International Journal of Data and Network Science, 8*(1), 431–438. https://doi.org/10.5267/j.ijdns.2023.9.012

[19] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95—International Conference on Neural Networks, 4,* 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

[20] Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence* (*Cat. No. 98TH8360*), 69–73. https://doi.org/10.1109/ICEC.1998.699146

[21] Shami, T. M., El-Saleh, A. A., Alswaitti, M., Al-Tashi, Q., Summakieh, M. A., & Mirjalili, S. (2022). Particle swarm optimization: A comprehensive survey. *IEEE Access, 10,* 10031–10061. https://doi.org/10.1109/ACCESS.2022.3142859

[22] James, J., Sandhya, L., & Thomas, C. (2013). Detection of phishing URLs using machine learning techniques. In *2013 International Conference on Control Communication and Computing,* 304–309. https://doi.org/10.1109/ICCC.2013.6731669

[23] Subasi, A., Molah, E., Almkallawi, F., & Chaudhery, T. J. (2017). Intelligent phishing website detection using random forest classifier. In *2017 International Conference on Electrical and Computing Technologies and Applications,* 1–5. https://doi.org/10.1109/ICECTA.2017.8252051

[24] Hota, H. S., Shrivas, A. K., & Hota, R. (2018). An ensemble model for detecting phishing attack with proposed remove-replace feature selection technique. *Procedia Computer Science, 132,* 900–907. https://doi.org/10.1016/j.procs.2018.05.103

[25] Jain, A. K., & Gupta, B. B. (2018). Towards detection of phishing websites on client-side using machine learning based approach. *Telecommunication Systems, 68*(4), 687–700. https://doi.org/10.1007/s11235-017-0414-0

[26] Joshi, A., & Pattanshetti, T. R. (2019). Phishing attack detection using feature selection techniques. In *Proceedings of International Conference on Communication and Information Processing.*

[27] Ubing, A. A., Jasmi, S. K. B., Abdullah, A., Jhanjhi, N. Z., & Supramaniam, M. (2019). Phishing website detection: An improved accuracy through feature selection and ensemble learning. *International Journal of Advanced Computer Science and Applications, 10*(1), 252–257. https://doi.org/10.14569/IJACSA.2019.0100133

[28] Mao, J., Bian, J., Tian, W., Zhu, S., Wei, T., Li, A., & Liang, Z. (2019). Phishing page detection via learning classifiers from page layout feature. *EURASIP Journal on Wireless Communications and Networking, 2019*(1), 43. https://doi.org/10.1186/s13638-019-1361-0

[29] Sahingoz, O. K., Buber, E., Demir, O., & Diri, B. (2019). Machine learning based phishing detection from URLs. *Expert Systems with Applications, 117,* 345–357. https://doi.org/10.1016/j.eswa.2018.09.029

[30] 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. https://doi.org/10.1108/EL-05-2019-0118

[31] Subasi, A., & Kremic, E. (2020). Comparison of adaboost with multiboosting for phishing website detection. *Procedia Computer Science, 168,* 272–278. https://doi.org/10.1016/j.procs.2020.02.251

[32] Ali, W., & Malebary, S. (2020). Particle swarm optimization-based feature weighting for improving intelligent phishing website detection. *IEEE Access, 8,* 116766–116780. https://doi.org/10.1109/ACCESS.2020.3003569

[33] Alsariera, Y. A., Adeyemo, V. E., Balogun, A. O., & Alazzawi, A. K. (2020). AI meta-learners and extra-trees algorithm for the detection of phishing websites. *IEEE Access, 8,* 142532–142542. https://doi.org/10.1109/ACCESS.2020.3013699

[34] Odeh, A., Keshta, I., & Abdelfattah, E. (2021). PHIBOOST—A novel phishing detection model using adaptive boosting approach. *Jordanian Journal of Computers Information Technology, 7*(1), 64–73. https://doi.org/10.5455/jjcit.71-1600061738

[35] Harinahalli Lokesh, G., & BoreGowda, G. (2021). Phishing website detection based on effective machine learning approach. *Journal of Cyber Security Technology, 5*(1), 1–14. https://doi.org/10.1080/23742917.2020.1813396

[36] Gupta, B. B., Yadav, K., Razzak, I., Psannis, K., Castiglione, A., & Chang, X. (2021). A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment. *Computer Communications, 175,* 47–57. https://doi.org/10.1016/j.comcom.2021.04.023

[37] Kang, M., & Tian, J. (2018). Machine learning: Data preprocessing. In M. G. Pecht & M. Kang (Eds.), *Prognostics and health management of electronics: Fundamentals, machine learning, and the Internet of Things* (pp. 111-130). Wiley. https://doi.org/10.1002/9781119515326.ch5

[38] Naganna, S. R., & Deka, P. C. (2014). Support vector machine applications in the field of hydrology: A review. *Applied Soft Computing, 19,* 372–386. https://doi.org/10.1016/j.asoc.2014.02.002

[39] Almaiah, M. A., Almomani, O., Alsaaidah, A., Al-Otaibi, S., Bani-Hani, N., Hwaitat, A. K. A., ..., & Aldhyani, T. H. H. (2022). Performance investigation of principal component analysis for intrusion detection system using different support vector machine kernels. *Electronics, 11*(21), 3571. https://doi.org/10.3390/electronics11213571

[40] Sharma, A. K., & Sahni, S. (2011). A comparative study of classification algorithms for spam email data analysis. *International Journal on Computer Science and Engineering, 3*(5), 1890–1895.

[41] Biau, G., & Scornet, E. (2016). A random forest guided tour. *TEST, 25*(2), 197–227, https://doi.org/10.1007/s11749-016-0481-7

[42] Rish, I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, 3*(22), 41–46.

[43] Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In *On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences CoopIS, DOA, and ODBASE 2003 Catania,* 986–996. https://doi.org/10.1007/978-3-540-39964-3_62

[44] Mamun, M. S. I., Rathore, M. A., Lashkari, A. H., Stakhanova, N., & Ghorbani, A. A. (2016). Detecting malicious URLs using lexical analysis. In *Network and System Security: 10th International Conference,* 467–482. https://doi.org/10.1007/978-3-319-46298-1_30

**How to Cite:** Almomani, O., Alsaaidah, A., Abu-Shareha, A. A., Alzaqebah, A., Almaiah, M. A., & Shambour, Q. (2025). Enhance URL Defacement Attack Detection Using Particle Swarm Optimization and Machine Learning. *Journal of Computational and Cognitive Engineering*, *4*(3), 296–308. https://doi.org/10.47852/bonviewJCCE52024668