Expert Systems With Applications 148 (2020) 113249

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer

Hadeel Alazzam[*], Ahmad Sharieh, Khair Eddin Sabri

Computer Science Department University of Jordan Amman, Jordan

ARTICLE INFO

Article history:
Received 19 August 2019
Revised 26 November 2019
Accepted 24 January 2020
Available online 25 January 2020

MSC:
00-01
99-00

Keywords:
Feature selection
Intrusion Detection System
KDDCUP
Pigeon Inspired Optimizer

ABSTRACT

Feature selection plays a vital role in building machine learning models. Irrelevant features in data affect the accuracy of the model and increase the training time needed to build the model. Feature selection is an important process to build Intrusion Detection System (IDS). In this paper, a wrapper feature selection algorithm for IDS is proposed. This algorithm uses the pigeon inspired optimizer to utilize the selection process. A new method to binarize a continuous pigeon inspired optimizer is proposed and compared to the traditional way for binarizing continuous swarm intelligent algorithms. The proposed algorithm was evaluated using three popular datasets: *KDDCUP*^{99}, NLS-KDD and UNSW-NB15. The proposed algorithm outperformed several feature selection algorithms from state-of-the-art related works in terms of TPR, FPR, accuracy, and F-score. Also, the proposed cosine similarity method for binarizing the algorithm has a faster convergence than the sigmoid method.

1. Introduction

Feature selection is the process of electing the most relevant features that contribute building a robust model (Liu & Motoda, 2012). Feature selection can be done manually or using several techniques and algorithms. It is an important step in building a robust Intrusion Detection System (IDS) to eliminate irrelevant features that raise false alarms and increases the accuracy of the system (Tang, Dai, & Xiang, 2019).

IDS is a software application or device that is intended to monitor the activities of network traffic to identify malicious contents or activities. An IDS is also designed in such a way that it issues alerts and reports upon the discovery of malicious content. Even though the majority of IDS systems are designed to detect and report suspicious network activity, it is noteworthy that there are advanced systems that have the capability of blocking suspicious network traffic (Scarfone & Mell, 2012).

Intrusion detection systems can be placed in two categories based on the detection method (Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee, & Karimipour, 2019). The first category is signature-based detection. It uses particular patterns within the network such as the sequence of bytes and then compares them with an existing database of signatures. The second one is anomaly-based detection system. This works by comparing the behavior of a network against an established baseline and is well suited to detect both known and unknown attacks.

An IDS deals with a large amount of data that includes false positives as well as irrelevant and redundant features. These features not only decrease the detection speed but also consume a lot of computational resources. It is essential to have a mechanism of selecting the best features to increase accuracy; training and testing speed (Zaman & Karray, 2009). Feature selection helps to solve some of the common problems in IDS through the identification of relevant features. As noted by (Mohammadi et al., 2019) relevant features carry essential information that greatly assists in the classification process. As an essential aspect worth noting about feature selection in IDS is that it reduces processing cost, minimizes storage space, and increases understanding of the test data.

Because feature selection is a machine learning concept, it is mostly implemented using a variety of algorithms. Feature selection is also accomplished using methods such as statistical analysis, support vector machine, neural networks, and data mining (Liu & Motoda, 2012). Additionally, feature selection assumes a detection mechanism that can be placed into three categories: random, incremental and decrement selection (Liu & Motoda, 2012). The selection mechanism is used to determine and select the relevant features in a dataset. It is noteworthy that feature selection can be achieved using various techniques including the use of intelligence patterns, swarm intelligence, artificial neural networks, deterministic algorithms, and fuzzy and rough sets (Maza & Touahria, 2018).

[*] Corresponding author.
E-mail addresses: hdy9160095@fgs.ju.edu.jo (H. Alazzam), sharieh@ju.edu.jo (A. Sharieh), k.sabri@ju.edu.jo (K.E. Sabri).

https://doi.org/10.1016/j.eswa.2020.113249
0957-4174/© 2020 Elsevier Ltd. All rights reserved.

Metaheuristics algorithms are mostly used for feature selection in intrusion detection systems especially due to their high level of accuracy (Al Shorman, Faris, & Aljarah, 2019). In this context, swarm intelligence is an important technique that is used in the implementation and categorization of metaheuristics algorithms. Swarm intelligence is a collective and artificial intelligence that is inspired by the behavior of insects and swarms. It is used to solve complex problems. Two popular approaches used in swarm intelligence include Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) (Chen, Zhou, & Yuan, 2019). Apart from the popularly utilized swarm-based feature selection algorithms, other custom-based algorithms utilize metaheuristics and swarm intelligence, and which have been used to develop various methodologies and frameworks.

The main Contributions of this paper are summarized as follows:

1. Summarize the state-of-the-art works related to feature selection algorithms for intrusion detection system.
2. Propose a novel feature selection algorithm for intrusion detection system based on pigeon inspired optimizer.
3. Propose a new methodology to binarize a continuous metaheuristic algorithm based on cosine similarity concept to make it suitable for discrete problems and compare it with the traditional way such as using sigmoid function to transfer the velocity to binary version.
4. Test the proposed feature selection algorithm and compare the result with six famous algorithms from the state-of-the-art works.

The rest of this paper is organized as follows: the state-of-the-art works are presented in Section 2. Section 3 introduces the Pigeon Intelligent Optimizer. The proposed feature algorithm is discussed in detail in Section 4, while Section 5 discusses the experimental results, and Section 6 concludes the paper.

2. Related works

Eesa, Orman, and Brifcani (2015) developed a feature selection model that utilizes the combination of ID3 classifier algorithm and bees algorithm. The model known as ID3-BA is designed to optimize the selection of the required subset of features in IDS. In this model, the bees algorithm is used for the generation of the required subset of features while the ID3 algorithm is used to construct the classifier. The ID3-BA model utilizes the KDD Cup99 (Knowledge Discovery and Data mining tools) dataset that contains 41 features for training and testing purposes. The proposed approach is evaluated using three different criterions: False Alarm Rate (FAR), Detection Rate (DR), and Accuracy. The experimental results indicate that ID3-BA produces high values of DR (91.02%), AR (92.002%), and lower values corresponding to FAR (3.917%). Further, the results indicate that using a subset of features instead of all features produces better classifications with regards to DR, AR and lower FAR.

A feature selection model that is based on a hybrid learning mechanism developed by (Keshtgary, Rikhtegar et al., 2018). The IDS mechanism combines feature selection and clustering. The former uses a support vector machine (SVM) while the latter uses K-Medoids clustering algorithm. Additionally, the approach also uses Nave Bayes classifier for the evaluation process using KDD CUP99 dataset. The proposed model is evaluated using three essential performance metrics including accuracy, detection rate, and alarm rates. The performance metrics are generated using true positives true negatives, false positives, and false negatives. The experimental results were compared with three other feature selection methods. The comparison methods included K-Medoids+GFR+ Nave Bayes, K-Medoids+ Nave Bayes, and 10-fold cross-validation + Nave Bayes (Keshtgary et al., 2018). The experimental results indicated that the proposed hybrid normalization approach produces better accuracy (91.5%), detection rate (90.1%), and false alarm rate (6.36%).

Emary, Zawbaa, and Hassanien (2016) developed a feature selection mechanism that is based on the principles of the binary grey wolf optimization. As the name suggests, the main purpose of the mechanism is to establish the optimal position of the relevant features during the classification process. The mechanism uses two approaches; stochastic crossover and a sigmoidal function for determining the updated grey wolf position. The strength of using the two approaches is to maximize the accuracy of the classification process as well as to minimize the number of selected features. The framework uses the dataset from the UCI (UC Irvine) repository (Asuncion & Newman, 2007) while the experimental results are compared to the results obtained using genetic algorithms and particle swarm optimizer. The evaluation criterion is based on the average selected feature, test accuracy, and statistical fitness. The experimental results indicate that the proposed method outperforms genetic algorithms and particle swarm optimizer in terms of search capability, selection fitness, and accuracy.

Authors in (Mohammadi et al., 2019) developed a methodology for IDS feature selection that uses a combination of a clustering algorithm that is implemented using filter and wrapper methods. The wrapper method uses linear correlation coefficient algorithm (FGLCC) while the filter method uses the cuttlefish algorithm (CFA). The proposed method uses a decision tree to construct the classifier while performance evaluation is based on the KDD CUP 99 dataset. During the experimental setup, performance evaluation was based on accuracy, detection rate, false positives, and fitness function. The experimental results are compared to the evaluation results obtained using 10Fold cross-validation and other feature-based algorithms. The evaluation results indicate that the proposed FGLCC-CFA combined algorithm produces a superior detection rate of 95.23%, an accuracy of 95.03%, and a false positive rate of 1.65%.

Another hybrid algorithm used a combination of filter-based and wrapper-based feature selection is proposed by (Ambusaidi et al., 2014). The filter-based method used to rank the features based on the principle of Mutual Information (MI), the aim behind the usage of filter-based is to find the most relevant features and eliminate the number of features entered to the wrapper that used Least Square Support Vector Machine (LSSVM) for the selection process. The proposed algorithm eliminates the number of features needed to train the model to only six features while achieves a higher TPR and lower FPR than filter-based techniques.

Zhou and Cheng (2019) managed to develop an IDS that uses an ensemble classifier for feature selection. The framework combines the use of the bat algorithm (BA) and correlation-based feature selection (CFS). Additionally, the ensemble classifier is built using Random Forest and Forest-based Penalizing Attributes. The experiment results use the CIC-IDS2017 dataset. The experiment uses various performance evaluation metrics including false positive rate, true positive rate, detection rate, precision rate, false alarm rate, and Matthews correlation coefficient. The results of the experiment were compared to those obtained from a similar approach albeit without a feature selection element. The results indicate that CFS-BA combined approach high accuracy results of 96.76%, a detection rate of 94.04%, and a low level of false alarms of 2.38%.

Acharya and Singh (2018) came up with a novel approach that uses the Intelligence Water Drops (IWD) algorithm for IDS feature selection. IWD is a bio-inspired algorithm that uses SVM for classifier construction. IWD is also regarded as a swarm intelligence optimization algorithm that uses metaheuristics. This approach was evaluated using the KDD CUP99 dataset with the performance criteria based on false alarms, detection rate, and accuracy. Further, the results of the experiments are compared to the existing approaches that use bio-inspired algorithms. The experimental results indicated that the IWD feature selection algorithm produces a high detection rate (91.35%), improved accuracy (93.12%) and a low rate of false alarms (3.35%).

Zorarpacı and Özel (2016) approach was motivated and designed using swarm intelligence optimization techniques. The approach uses the hybrid combination of the differential evolution algorithm and the artificial bee colonization technique. The evaluation and classification processes are done using fifteen datasets obtained from the UCI repository. The evaluation process also included a comparison of the results obtained from the hybrid approach to those obtained from other feature selection approaches. The methods used for feature selection comparison included the chi-square, information gain, as well as CFS (correlation feature selection). Performance criteria were based on F-measure (best, worst, and average), accuracy, performance, and detection rate and tested using 10-folds classification validation. The simulation results indicated that the hybrid approach managed to yield a high-performance rate as well as a high rate of classification accuracy.

Another related work that is motivated by the concept of swarm intelligence and metaheuristics was developed by (Enache & Sgarciu, 2014). The approach is aimed at enhancing and improving the already existing techniques that use SVM. The developed methodology by (Enache & Sgarciu, 2014) involves the combination of support vector machine and the Bat algorithm. The feature selection mechanism is based on the concepts of Levy flights and the Binary Bat Algorithm. Further, the experimental setup uses a test model based on the NSL-KDD data set (Tavallaee, Bagheri, Lu, & Ghorbani, 2009). The model was implemented in IDS with the performance results indicating an attack detection rate of 95.05%, an accuracy of 90.06%, and a false alarm rate of 4.4%.

Another feature selection algorithm for IDS was proposed by (Chung & Wahid, 2012). The proposed algorithm introduces a new simplified version of PSO for feature selection named Simplified Swarm Optimization (SSO) that composed a local search strategy to accelerate the feature selection process by discovering the best neighboring solution. The proposed algorithm reduces the number of features needed to represent the behavior of network traffic from *KDDCUP*^{99} from 41 to only 6 features and achieved better accuracy than the standard PSO with a 93.3% accuracy.

Finally, (Ruggieri, 2002) work is motivated by the concepts of the firefly algorithm. The firefly algorithm is used to deploy the filter and the wrapper used in the feature selection process. In this context, the firefly algorithm is a metaheuristic optimization technique that is inspired by the behavior of fireflies. The experimental approach is tested and evaluated using the *KDDCUP*^{99} datasets and subjected to Bayesian Network and C4.5 features (Ruggieri, 2002). The experimental comparison was performed using the F-measure obtained from using 10 and 4 features. The experimental results from the simulation of a Denial of Service (DoS) attack show an improvement of the accuracy at 99.98% and a false positive rate of 0.01%.

Table 1 presents a summary of related works that proposed metaheuristic or hybrid approaches of features selection for IDS with the dataset used to evaluate their works and the Number of Feature (NF) used for training their model. The - sign indicate that the author didn't report the NF for the full dataset.

To sum up, it is notable from the literature that the proposed feature selection algorithms deal with a tradeoff between the TPR and FPR, and there is no particular number of features or subset of features that has been agreed by the literature works. The proposed approach uses a fitness function that consider TPR, FPR and the number of features to select the best subset of features based on these three important metrics, in spite of the related works that used two of these metrics to define the fitness function and ignored the number of features. Another challenge of using optimization problem for feature selection is the time complexity. The Pigeon optimizer is considered as the best global convergence among other swarm intelligence algorithms (Yi, Wen, & Li, 2016). In this paper, a pigeon inspired optimizer for feature selection is proposed for IDS. Also, a new method for discretizing a continuous problem based on cosine similarity that aims to accelerate the process of convergence is proposed in this paper.

3. Continuous Pigeon Inspired Optimizer

Pigeon Inspired Optimizer (PIO) is one of the new developed bio-inspired swarm intelligence algorithms (Duan & Qiao, 2014). Swarm intelligence algorithms have been adopted by researchers to solve optimization problems. Swarm intelligence used to solve Non-deterministic Polynomial (NP) problems or when the search space is too large. It mimics the social organisms of this swarm using the base of learning by trying to enhance the quality of the solutions using a mathematical model based on the natural behavior of these swarms to adopt the position and the velocity of the individuals (Shi et al., 2001).

Pigeons are very popular kind of birds that have the ability to fly long distance to search for food. Also, pigeons have special homing behavior that widely used during the first and second world wars where the carrier pigeons were used to carry messages (Varun & Kumar, 2018).

Pigeons derived their homing behavior from two main operators: map and compass and landmark operators. Some investigation of pigeon homing skills clarified that the pigeon abilities to navigate its homeland come from tiny magnetic particles located in its peak. These particles send signals to its brain through the trigeminal nerve (Varun & Kumar, 2018).

The homing skills of pigeons can be expressed mathematically based on two main operators the map and compass and landmark operators.

Pigeons used their ability of magneto reception to sense the magnetic field of the earth, besides that pigeons can perceive the suns altitude as a compass to adjust their direction. Whenever the pigeons become closer to their destination, they become less depending on the map and compass operator (Duan & Qiao, 2014).

To simplify this operator, it can be mathematically expressed by adjusting the position X_i and velocity V_i of pigeon i in each iteration. The values of X_i and V_i are updated for the next iteration $(t+1)^{th}$ depend on the value of the current iteration $t^t hin$ Eqs. 1 and 2 (Duan & Qiao, 2014).

$$V_i(t+1) = V_i(t).e^{-Rt} + rand.(X_g - X_i(t)) \tag{1}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{2}$$

Where R is a map and compass factor, while *rand* is a uniform random number in the range [0, 1], X_g is the global best solution, $X_i(t)$ denotes the current position of a pigeon at instance t, and $V_i(t)$ denotes the current velocity of a pigeon at iteration t.

As shown in Fig. 1, all pigeons adjust their flying position according to map and compass operator by following the best pigeon position. All of pigeons position is evaluated by a particular objective function. The best pigeon in Fig. 1 is represented by a black pigeon, all other pigeon will follow this pigeon according to Eq. 1, where the first part of the equation represents the current direction of the pigeon and is illustrated in Fig. 1 by a thin straight arrow, while the second part of the Eq. 1 represents the direction of the best pigeon and illustrated in Fig. 1 by a thick arrow. The summation of these two vectors is the next flying direction for the pigeon. All pigeons will adjust their position according to the new direction calculated in Eq. 1 and 2.

Table 1
Summary of related works and some performance: Detection Rate (DR) and False Positive Rate (FPR).

Reference	Algorithm	NF	Dataset	DR%	FPR%
(Chung & Wahid, 2012)	SSO	6	*KDDCUP*^{99}	-	-
(Eesa et al., 2015)	ID3-BA	5	*KDDCUP*^{99}	91.02%	3.9%
(Enache & Sgarciu, 2014)	BAT	18	NSL-KDD	95.05%	4.4%
(Ambusaidi et al., 2014)	LSSVM	6	*KDDCUP*^{99}	-	-
(Aslahi-Shahri et al., 2016)	GA	10	*KDDCUP*^{99}	97.03%	1.7%
(Acharya & Singh, 2018)	IWD	9	*KDDCUP*^{99}	91.35%	3.35%
(Keshtgary et al., 2018)	SVM	10	*KDDCUP*^{99}	90.1%	6.36%
(Selvakumar & Muneeswaran, 2019)	Firefly	10	*KDDCUP*^{99}	-	-
(Mohammadi et al., 2019)	Cuttlefish	10	*KDDCUP*^{99}	95.23%	1.65%
(Zhou & Cheng, 2019)	CSF-BA	-	CICIDS2017	94.04%	2.38%

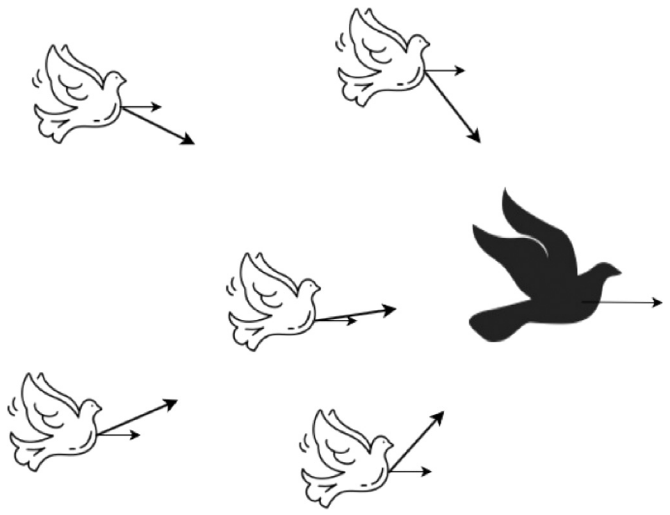
Fig. 1. White pigeons adjust their flying position according to map and compass operator by following the best pigeon (black) position.

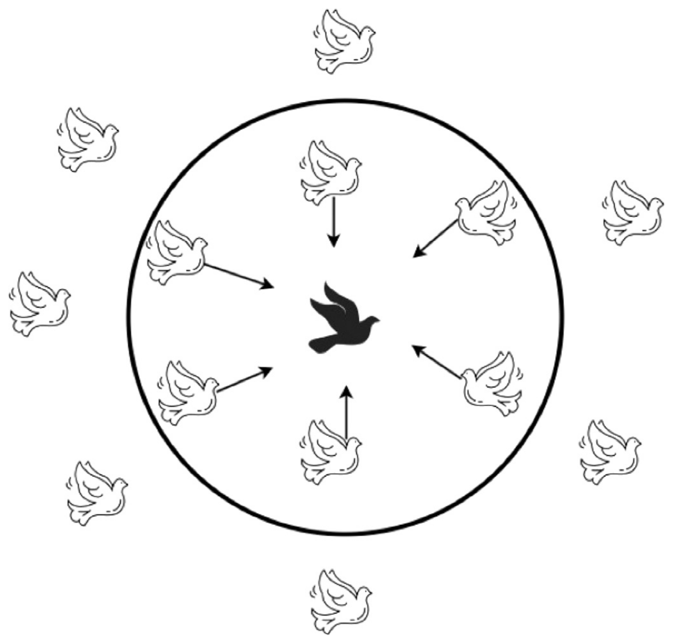
Fig. 2. Landmark operator: wherein computer simulation, the desirable destination position is represented by a black pigeon, while the pigeons in the circle are a half number of pigeons.

In landmark operator, all the pigeons are ranked according to their fitness value. In each generation, the number of pigeons is updated by Eq. 3, where only half number of pigeons is considered to calculate the desired position of the centered pigeon, while all other pigeons adjust their destination by following the desirable destination position. The position of the desired destination is calculated by Eq. 4, while all other pigeons update their position toward this position by Eq. 5 (Duan & Qiao, 2014).

Fig. 2 shows the landmark operator, wherein computer simulation, it is assumed that the pigeons are always not familiar with their landmark and have to follow up a desirable destination position from the top-ranked pigeons (Duan & Qiao, 2014). The desirable destination position is represented by a black pigeon in Fig. 2, while the pigeons in the circle are the half number of pigeons calculated by Eq. 3.

$$N_p(t+1) = \frac{N_p(t)}{2} \tag{3}$$

Where N_p is the number of pigeons in the current iteration t.

$$X_c(t+1) = \frac{\sum X_i(t+1).Fitness(X_i(t+1))}{N_p \sum Fitness(X_i(t+1)} \tag{4}$$

Where X_c is the position of the centered pigeon (desired destination), while X_i is the current position of all pigeons.

$$X_i(t+1) = X_i(t) + rand.(X_c(t+1) - X_i(t)) \tag{5}$$

Algorithm 1 shows an overall procedure for PIO in general. As the algorithm shows that there are two main while loops. The first while loop considers the map and compass operator, and the sec-

Algorithm 1 Continuous Pigeon Inspired Optimizer (CPIO).

Input: N_p, Space Dimension D, Map and compass factor R, Number of iterations nc_1, nc_2 where $nc_1 > nc_2$.
Output: Global Solution X_g

```
1: Initialize $X_i$ for each Pigeon randomly.
2: Evaluate Pigeons $(X_1, X_2, \ldots, X_{N_p})$ by their fitness values.
3: Xg = best pigeon (minimum fitness)
4: **while** $(n_c >= 1)$ **do**
5:     Update velocity and path for each pigeon by equations 1 and 2.
6:     Evaluate Pigeons $(X_1, X_2, \ldots, X_{N_p})$ by their fitness values.
7:     Update $X_g$
8: **end while**
9: **while** $(N_p >= 1)$ **do**
10:    Sort pigeons by their fitness values.
11:    $Np = N_p/2$
12:    Calculate desired destination by Equation 4
13:    Update pigeon position by equation 5.
14:    Update $X_g$
15: **end while**
```

Table 2
PIO map to feature selection optimization problem.

PIO Concept	Feature Selection representation
Number of Pigeons N_p	Number of solutions
Position of Each Pigeon X_p	Solution (selected features)
Pigeon or solution (X_p) length	Total number of features
Best Pigeon	The solution that has the best fitness value
The velocity of each Pigeon V_p	The amount of change toward the best pigeon
Fitness Function	Model Evaluation based on TPR, FPR, and number of selected features
N_c	Number of iterations

ond loop starts after the first one end in order to assess their route and make corrections.

4. PIO for feature selection

Recently, PIO algorithm proved its effectiveness in solving many optimization problems such as air robot path planning (Duan & Qiao, 2014), three-dimensional path planning (Zhang & Duan, 2015), automatic landing system (Deng & Duan, 2016), and PID design controller (Sun & Duan, 2014). In this paper, we adapt a feature selection algorithm for IDS based on a new binary version of Pigeon Inspired Optimizer. In this section, two versions of PIO are proposed. The first version or algorithm uses the sigmoidal function to discretize the velocity of pigeons, while the second version proposes a new modified binary version of the base PIO that uses the cosine similarity to define the velocity of the pigeons. Both versions used the same fitness function, on the other hand, each version has its a different way for pigeon or solution representation. Table 2 represents the mapping process of PIO to feature selection optimization problem.

4.1. Fitness function

Fitness function or objective or cost function are the terminology for a procedure to evaluate the fitness of the solutions.The fitness function evaluates the solution which is a subset of selected features in terms of True Positive Rate (TPR), False Positive Rate (FPR) and number of features. The number of features is included in the fitness function so if there is any feature exists but does not affect the TPR or the FPR (quality of the solution), we prefer to eliminate it. Eq. 6 present the formula used to evaluate the pigeon or solution fitness.

$$FF = w_1 * \frac{SF}{NF} + w_2 * FPR + w_3 * \frac{1}{TPR} \tag{6}$$

Where *SF* is the number of selected features, *NF* is the total number of features and $w_1 + w_2 + w_3 = 1$. The values of the weights were set as follow, $w_1 = 0.1$,and $w_2 = w_3 = 0.45$ since TPR and FPR are equally important (Gupta, Joshi, Bhattacharjee, & Mundada, 2012). For $KDDCUP^{99}$ dataset the total number of features is 41 (Tavallaee et al., 2009).

4.2. Sigmoid_PIO for feature selection

The first version of the proposed PIO feature selection defines the solution or pigeon by a vector with length equals to the number of features; in case of KDD CUP, the pigeon or solution vector length is 41. As the PIO base procedure deals with the position of pigeon in a continuous manner, the proposed PIO for feature selection defines the solution representation as a vector of a particular length (number of features), where the values of the position and velocity vectors initially are random numbers between [0, 1].

A traditional way is used to calculate the velocity of each pigeon by equation (1), then a sigmoidal function was used to transfer the velocity into binary version by Eq. 7 as proposed by (Too, Abdullah, & Mohd Saad, 2019). For binarize a swarm intelligent algorithm, each pigeon position will be updated according to the sigmoid function value and based on probability of a random uniform number between [0, 1] by Eq. 8. The rest of the algorithm will be the work as the traditional PIO except the update position in the landmark operator. Also, the sigmoid function will be used to transfer the velocities, then the positions will be updated according to it. Fig. 3 shows the overall design for Sigmoid_PIO feature selection.

$$S(V_i(t)) = \frac{1}{1 + e^{\frac{-v_i}{2}}} \tag{7}$$

$$X(t)_{(i,p)}[i] = \begin{cases} 1, & if(S(V_i(t)) > r) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Where $V_i(t)$ is the pigeon velocity in iteration t and r is a uniform random number.

4.3. Modified binary PIO for feature selection (Cosine_PIO)

The proposed modified binary PIO is designed to overcome the limitation occurred by the first proposed approach. Cosine_PIO used the cosine similarity to calculate the velocity of the pigeons. The Cosine_PIO differs from the Sigmoid_PIO by three points: pigeon or solution representation, the calculation of new position and velocity since this version is binarized, and the proposed Cosine_PIO adds a new feature to the base PIO that allows new blood of pigeon to join the existing one under particular conditions. This enhancement increases the opportunity of reaching the optimal solution.

4.3.1. Pigeon or solution representation

The solution in Cosine_PIO is a vector with length of the number of inputs (number of features). The value of the solution is initialized by randomly binary value zero or one. Zero value indicates that the corresponding feature is absent in the current solution while the value one indicates the presence of the corresponding feature in the solution. Fig. 4 shows an example of a randomly generated solution for $KDDCUP^{99}$ dataset.

4.3.2. Modified map and compass operator

As mentioned earlier PIO base algorithm designed for a continuous problem. The map and compass operator are the main procedure for updating the pigeon position based on the velocity and the position of the best pigeon in the swarm. The base PIO works by subtracting the position Xi of the local pigeon from the global or best pigeon as declared in Eq. 1. But, in binary PIO, we cannot subtract the binarized vector as regular subtraction. New equations were used to simulate the subtraction process and to update the pigeon position X_p and velocity toward the global pigeon X_g. Eq. 9 presents the calculation of the pigeon velocity. The velocity here depends on the amount of similarity between the solutions (each pigeon and the global solution), so every pigeon or solution has a different velocity value. The calculation of velocity is based on the cosine similarity formula to find the similarity ratio

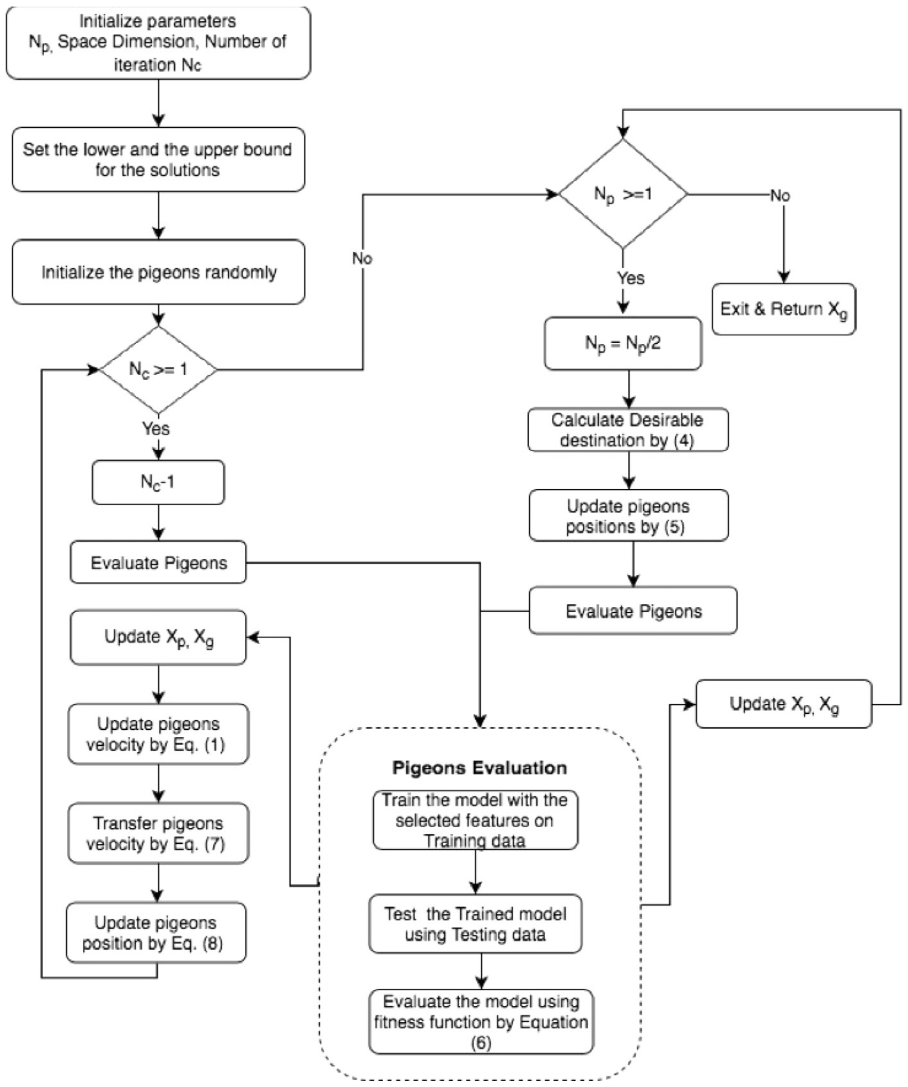
Initialize parameters N_p, Space Dimension, Number of iteration Nc
Set the lower and the upper bound for the solutions
Initialize the pigeons randomly
N_c >= 1
Yes
No
N_c-1
Evaluate Pigeons
Update X_p, X_g
Update pigeons velocity by Eq. (1)
Transfer pigeons velocity by Eq. (7)
Update pigeons position by Eq. (8)
N_p >=1
Yes
No
Exit & Return X_g
N_p = N_p/2
Calculate Desirable destination by (4)
Update pigeons positions by (5)
Evaluate Pigeons
Update X_p, X_g
Pigeons Evaluation
Train the model with the selected features on Training data
Test the Trained model using Testing data
Evaluate the model using fitness function by Equation (6)

Fig. 3. Sigmoid_PIO feature selection design.

0	1	1	0	1	...	0
1	2	3	4	5	...	41

Fig. 4. Pigeon representation for *KDDCUP*99 Dataset.

between the local pigeon X_p and the global one X_g. Based on the velocity value V_p from Eq. 9 the position of the pigeon will be updated by Eq. 10. According to Eq. 10, the position of the pigeon will be updated according to the probability of its similarity to the global solution.

$$V_p = Cosine\,Similarity(X_g, X_p) = \frac{X_g.X_p}{||X_g||.||X_p||} = \frac{\sum_{i=0}^{n-1} X_{p,i}X_{g,i}}{\sqrt{\sum_{i=0}^{n-1} {X_{p,i}}^2}\sqrt{\sum_{i=0}^{n-1} {X_{g,i}}^2}} \tag{9}$$

$$X(t)_{(i,p)}[i] = \begin{cases} X(t-1)_p[i], & if(S(V_i(t)) > r) \\ X(t-1)_g[i], & \text{otherwise} \end{cases} \tag{10}$$

Where r is a uniform random number.

Based on Eq. 10, if the solution is not neighbor to the global solution, then the probability of updating its position toward the global solution is higher than the probability if the current solution is a neighbor to the global solution.

4.3.3. Modified landmark operator

The first part of the landmark operator which computes the desirable pigeon destination is the same as the base one. All the pigeons are ranked according to their fitness value. In each generation, the number of pigeons is updated by Eq. 3, where only half number of pigeons is considered to calculate the desired position of the centered pigeon, while all other pigeons adjust their destination by following the desirable destination position. The position of the desired destination is calculated by Eq. 4.

The second part of the landmark operator where all pigeons update their positions toward the desired destination is different as the desired destination is a binary vector. Thus, all the pigeons will update their positions initially by calculating their velocity by Eq. 9, and then update their positions according to Eq. 10.

4.3.4. Entering new pigeon

Another modification applied on the binary PIO for feature selection is the possibility that a new pigeon joins the pigeons' swarm. The idea of this procedure comes from the high possibility of replicated solutions or pigeons can be found in binary PIO. The new pigeon entrance can be done only in map and compass

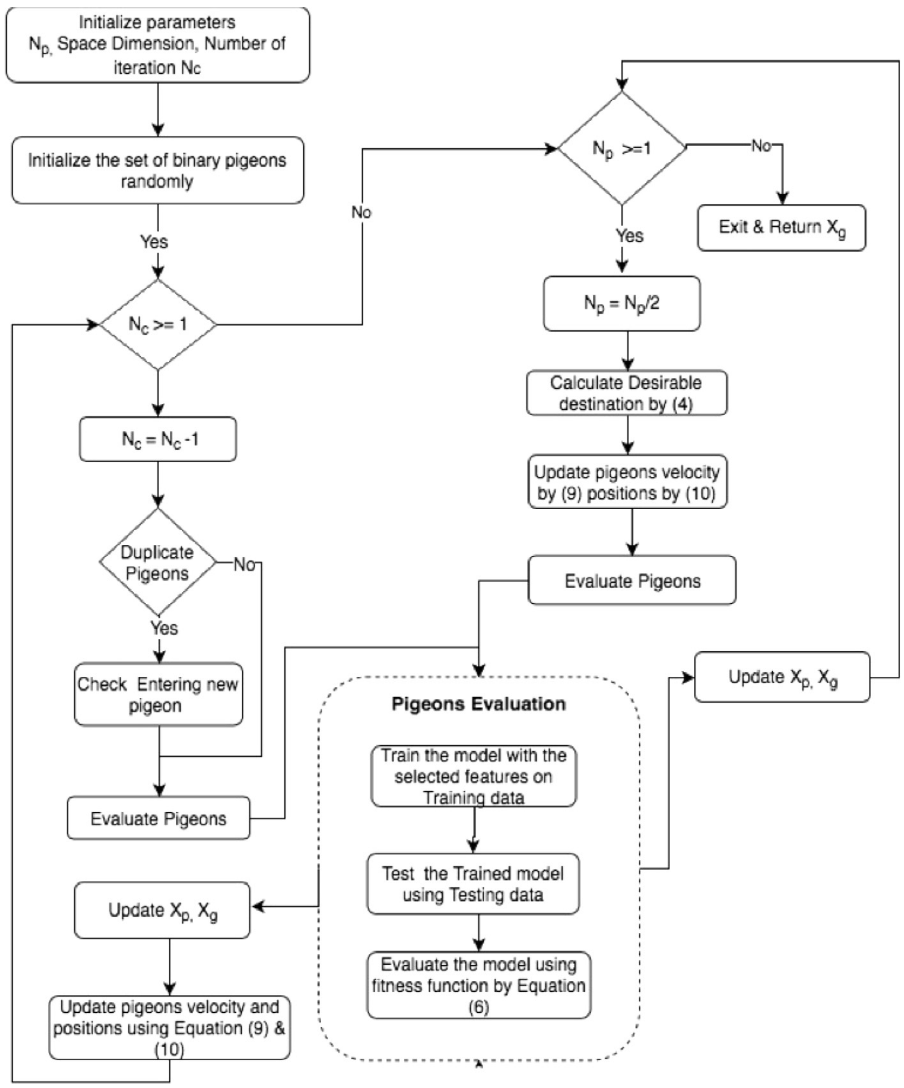
Initialize parameters N_p, Space Dimension, Number of iteration Nc
Initialize the set of binary pigeons randomly
Yes
$N_c >= 1$
No
$N_c = N_c - 1$
Duplicate Pigeons
No
Yes
Check Entering new pigeon
Evaluate Pigeons
Update X_p, X_g
Update pigeons velocity and positions using Equation (9) & (10)
$N_p >= 1$
No
Exit & Return X_g
Yes
$N_p = N_p/2$
Calculate Desirable destination by (4)
Update pigeons velocity by (9) positions by (10)
Evaluate Pigeons
Update X_p, X_g
Pigeons Evaluation
Train the model with the selected features on Training data
Test the Trained model using Testing data
Evaluate the model using fitness function by Equation (6)

Fig. 5. Modified binary Cosine_PIO feature selection design.

operator. If there are replicated solutions, a judge must be taken to replace the replicate pigeon with a new neighbor solution by altering 0.2 of the current solution randomly to join the swarm. This judge will be based on a uniform random number, if the number is higher than 0.5 then a new pigeon will join the swarm otherwise discard it. This will help in exploiting the search space. Fig. 5 illustrates the mechanism of the feature selection based on modified binary PIO.

In summary, the discrete pigeon for feature selection is a powerful algorithm, however the binaries process limits the effectiveness of the landmark operator. Since the landmark operator works by calculating the positions of the desirable pigeon destination as an average of high fitness pigeons. Thus, the probability of producing a new better solution from the average of binary vectors will be low. While the landmark operator makes an effective enhancement on the solution when applied on a continuous problem (Duan & Qiao, 2014).

5. Experiments and results

In this section, the dataset used for evaluation is introduced. The main categories of attacks are listed and the distribution of the training dataset is discussed. Also, the development model is described in detail with the main preprocessing steps. This section introduces the performance metrics used to evaluate the proposed approach and finally, the conducted experiments are discussed.

5.1. Dataset

Three popular datasets have been used to evaluate the proposed feature selection algorithm: *KDDCUPP*99 (Lippmann et al., 1998), NSL-KDD (Ahmed, Mahmood, & Hu, 2016; Revathi & Malathi, 2013) and UNSW-NB15 (Moustafa & Slay, 2015).

5.1.1. DAPRA KDDCUP99 Dataset

DARPA dataset was initially developed in 1998 with the aim of improving off-line intrusion detection (Lippmann et al., 1998). Additionally, 1998 DARPA dataset was developed with the aim of improving research and survey in the field of intrusion detection. *KDDCUP*99 is an improved version of 1999 DARPA dataset that is used for the development of an intrusion detection system with the aim of differentiating between bad and good connections.

The dataset is primarily designed for the detecting intrusions in a network through a simulation mechanism in a military environment. *KDDCUP*99 was designed and developed using DARPA98 IDS and is used to simulate four different types of attacks(Tavallaee et al., 2009). The attacks can be classified into four main categories (Stolfo, Fan, Lee, Prodromidis, & Chan, 2000):

Table 3
KDDCUP^{99}, NSL-KDD Features with their data type and category.

Category	No.	Name	Data Type	Category	No.	Name	Data Type
Basic	1	duration	continuous	**Content**	22	is_guest_login	symbolic
	2	protocol_type	symbolic		23	count	continuous
	3	service	symbolic		24	srv_count	continuous
	4	Flag	symbolic		25	serror_rate	continuous
	5	src_bytes	continuous		26	srv_serror_rate	continuous
	6	dst_bytes	continuous		27	rerror_rate	continuous
	7	Land	symbolic		28	srv_rerror_rate	continuous
	8	wrong_fragment	continuous		29	same_srv_rate	continuous
	9	urgent	continuous		30	diff_srv_rate	continuous
Content	10	Hot	continuous		31	srv_diff_host_rate	continuous
	11	num_failed_logins	continuous	**Traffic**	32	dst_host_count	continuous
	12	logged_in	symbolic		33	dst_host_srv_count	continuous
	13	num_compromised	continuous		34	dst_host_same_srv_rate	continuous
	14	root_shell	continuous		35	dst_host_diff_srv_rate	continuous
	15	su_attempted	continuous		36	dst_host_same_src_port_rate	continuous
	16	num_root	continuous		37	dst_host_srv_diff_host_rate	continuous
	17	num_file_creations	continuous		38	dst_host_serror_rate	continuous
	18	num_shells	continuous		39	dst_host_srv_serror_rate	continuous
	19	num_access_files	continuous		40	dst_host_rerror_rate	continuous
	20	num_outbound_cmds	continuous		41	dst_host_srv_rerror_rate	continuous
	21	is_host_login	symbolic		42	Class	symbolic

Table 4
Corrected *KDDCUP*^{99} Training Dataset Distribution.

	# of Instances	Percentage %
Normal	97,277	19.69%
DOS	391,458	79.24%
Probe	4107	0.83%
R2L	1126	0.23%
U2L	52	0.01%
Total	494,019	100%

- **Denial of Service Attacks (DoS)**: It is an attempt by the attacker to restrict network usage by disrupting service availability to the intended users.
- **User to Root Attacks (U2R)**: Occurs when the attacker has access from a normal user account and tries to gain root access through system vulnerabilities.
- **Remote to Local attacks (R2L)**: The attacker does not have an account on local system but tries to gain access through sending network packets to exploit the vulnerabilities and gain access as a local user.
- **Probing attacks**: Occurs when the attacker scans the system network to collect information about the system in aims to use it for avoiding the system security control.

KDDCUP^{99} dataset contains 4,898,431 and 311,029 connections in Training and Testing set respectively (Tavallaee et al., 2009). However, this paper experiments are based on 10% of the corrected Training and Testing set. It is important to mention that the testing dataset contains new types of attacks not exists in the training set. The training dataset contains 24 types of attacks while the test set contains an additional 14 attacks. The *KDDCUP*^{99} has 41 features that can be grouped into three major categories: basic features, traffic features, and content features. Table 3 presents the set of features in *KDDCUP*^{99} and the NSL-KDD datasets with the corresponding group and the datatype (continuous or symbolic). While Table 4 presents the distribution of 10% corrected training dataset of the *KDDCUP*^{99}that contains 494,019 connection records.

5.1.2. NSL-KDD

NSL-KDD dataset is an improved version of KDDCUP'99. It has been suggested to solve some problems of the *KDDCUP*^{99} that mentioned by (Tavallaee et al., 2009). NSL-KDD has a reasonable number of records in its training and testing tests. It has the same features of the original *KDDCUP*^{99}. It is an effective benchmark for researchers to compare their proposed IDSs (Revathi & Malathi, 2013).

5.1.3. UNSW-NB15 Dataset

The UNSW-NB15 dataset was developed by IXIA PerfectStorm. It's used to simulate and generate both real and contemporary attack models. It is a tool known as Tcpdump, which contains up to 100 GB of Pcap files used to simulate nine different types of attacks. The attacks include DOS, ShellCode, Worms, Fuzzers, Backdoors, Exploits, Analysis, Generic, and Reconnaissance. Additionally, the dataset is composed of twelve algorithms that are used for the generation of 49 features belonging to the class label (Moustafa & Slay, 2015). Table 5 presents the set of features in UNSW-NB15 with the corresponding group and the datatype.

5.2. Data preprocessing

Data preprocessing composed of three main steps: Label Transfer and Data Transfer, Remove duplication, and Data normalization. In label transfer and data transfer, all the symbolic data are transferred to numeric values. Also, the class column inputs are transformed to binary classes 0 or 1, where 0 indicate normal record while 1 indicate an attack record in spite of the attack type.

It is important to remove duplicate records in the training set to avoid the classifiers to be biased to most frequent records and prevent it from learning infrequent records such as U2L attack (Alazzam, Alsmady, & Shorman, 2019). To avoid this, duplicate records of *KDDCUP*^{99} were removed and the number of records in the training set after eliminating the redundant data is 145,584 instances (Basaran, Ntoutsi, & Zimek, 2017). Both NSL-KDD and UNSW-NB15 datasets do not have any duplicate records.

Normalizing the data is an important step to eliminate the biased with the features of larger values from the dataset (Sahu, Sarangi, & Jena, 2014). Data normalization is the process of transforming or scaling the data values of each feature into a proportional range. The used dataset was normalized into the range [0, 1] according to Eq. 11 (R. Al Shorman, Faris, Castillo, Merelo Guervs, & Al-Madi, 2018).

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{11}$$

Table 5
UNSW-NB15 Features with their data type and category.

Category	No.	Name	Data Type	Category	No.	Name	Data Type
Flow	1	srcip	nominal	**Content**	25	trans_depth	integer
	2	sport	integer		26	res_bdy_len	integer
	3	dstip	nominal	**Time**	27	Sjit	Float
	4	dsport	integer		28	Djit	Float
	5	proto	nominal		29	Stime	Timestamp
Basic	6	state	nominal		30	Ltime	Timestamp
	7	dur	Float		31	Sintpkt	Float
	8	sbytes	Integer		32	Dintpkt	Float
	9	dbytes	Integer		33	tcprtt	Float
	10	sttl	Integer		34	synack	Float
	11	dttl	Integer		35	ackdat	Float
	12	sloss	Integer	**General**	36	is_sm_ips_ports	Binary
	13	dloss	Integer	**Purpose**	37	ct_state_ttl	Integer
	14	service	nominal		38	ct_flw_http_mthd	Integer
	15	Sload	Float		39	is_ftp_login	Binary
	16	Dload	Float		40	ct_ftp_cmd	integer
	17	Spkts	integer	**Connection**	41	ct_srv_src	integer
	18	Dpkts	integer		42	ct_srv_dst	integer
Content	19	swin	integer		43	ct_dst_ltm	integer
	20	dwin	integer		44	ct_src_ ltm	integer
	21	stcpb	integer		45	ct_src_dport_ltm	integer
	22	dtcpb	integer		46	ct_dst_sport_ltm	integer
	23	smeansz	integer		47	ct_dst_src_ltm	integer
	24	dmeansz	integer		48	attack_cat	nominal
					49	Class	binary

5.3. Feature selection

KDDCUP^{99} and NSL-KDD datasets contain 41 features, while UNSW-NB15 dataset contains 49 features, not all features are important to build IDS. A subset of those features must be selected to achieve high detection rate and low false alarms. Moreover, the feature selection process is important to eliminate the number of features that are necessary to build IDS. In this paper, an adopted metaheuristic to manage feature selection process based on PIO is proposed.

5.4. Classifier training and testing

Once the feature selection process is accomplished using the proposed feature selection algorithm, the set of features is trained using a particular classifier. The classifier aims to distinguish between normal and attacks classes. Then, the trained model is evaluated using a testing set. In this paper, decision tree (DT) is used to train and evaluate the subset of features recommended by the proposed feature selection and will be used to compare several proposed feature selection algorithms from state-of-the-art related works.

5.5. Performance metrics

There are several metrics to evaluate feature selection algorithms. The selected measure is depending on the nature of the application. Most researches evaluate their IDS using True Positive Rate (TPR) and the False Positive Rate (FPR) performance metrics. In this section, we define the set of performance metrics used to evaluate the proposed approach. All of the selected metrics can be calculated using the confusion matrix output. A confusion matrix is represented by four major parameters (Kumar, 2014):

- **True Positive (TP)**: Number of Attack instances classified correctly.
- **True Negative (TN)**: Number of Normal instances classified correctly.
- **False Positive (FP)**: Number of Normal instances wrongly classified as an attack.
- **False Negative (FN)**: Number of Attack instances wrongly classified as normal.

Performance metrics definition and formulas according to the confusion matrix as following (Shewale & Patil, 2016):

- **Sensitivity** (True Positive Rate (TPR), Detection Rate or Recall): Measures the proportion of actual attacks that are correctly identified as in Eq. 12.

$$TPR = \frac{TP}{TP + FN} \tag{12}$$

- **Accuracy**: Measures the proportion of correct classified classes to the total number of classifications as in Eq. 13.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

- **False Positive Rate** (FPR or False Alarms): Measures the proportion of normal that are identified as attacks as in Eq. 14.

$$FPR = \frac{FP}{TN + FP} \tag{14}$$

- **F-score** (F-measure): Measure the accuracy of the model by considering both precision and recall as in Eq. 15.

$$F - Score = \frac{2 * TP}{2 * TP + FP + FN} \tag{15}$$

5.6. Results

In this section, the PIO feature selection algorithm evaluated using three datasets: *KDDCUP*^{99}, NSL-KDD and UNSW-NB15. The algorithm compared to some of the state-of-the-art feature selection algorithms, such as GA, PSO and BAT for network intrusion detection system. All of the feature selection algorithms are evaluated using DT classifier from scikit-learn library in python (Pedregosa et al., 2011), since DT can easily handle features interaction compared to other base classifiers (Peddabachigari, Abraham, & Thomas, 2004). All data preprocessing steps were applied to all examined algorithms to achieve fairness. According to this, the experimental results may vary from the one reported in related works.

Table 6
Selected set of features from *KDDCUP*99 by several feature selection algorithms.

Reference	Technique	# of features	Selected set of features
(Chung & Wahid, 2012)	SSO	6	[3, 5, 6, 27, 33, 35]
(Ambusaidi et al., 2014)	LSSVM	6	[3, 5, 23, 32, 34, 35]
(Aslahi-Shahri et al., 2016)	GA	10	[2, 3, 4, 8, 17, 21, 23, 31, 34, 36]
(Keshtgary et al., 2018)	SVM	10	[2, 3, 4, 5, 6, 8, 13, 22, 23, 24]
(Mohammadi et al., 2019)	Cuttlefish	10	[4, 10, 13, 22, 23, 24, 29, 35,36, 41]
(Mohammadi et al., 2019)	FGLCC	16	[4, 6, 10, 13, 22, 23, 24, 27, 29, 30, 32, 35, 36, 39, 40, 41]
Proposed Approach	Sigmoid_PIO	10	[3, 4, 6, 11, 13, 18, 23, 36, 37, 39]
Proposed Approach	Cosine_PIO	7	[3, 4, 6, 13, 23, 29, 34]

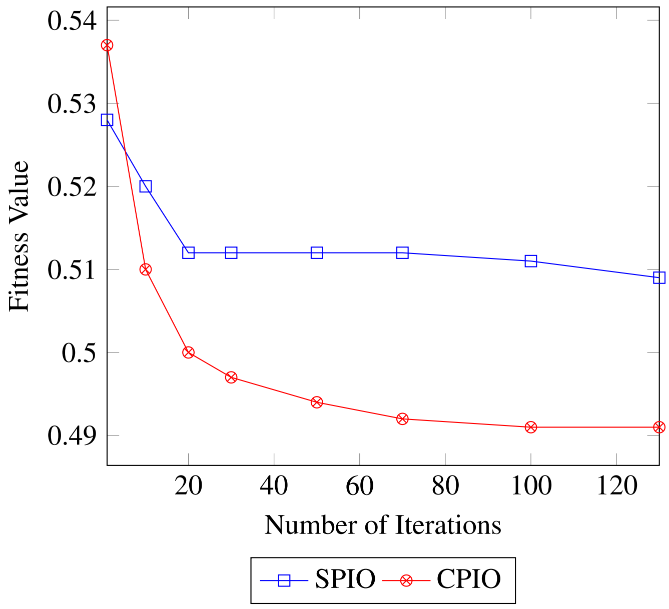

Fig. 6. Convergence curve of proposed approaches on *KDDCUP*99.

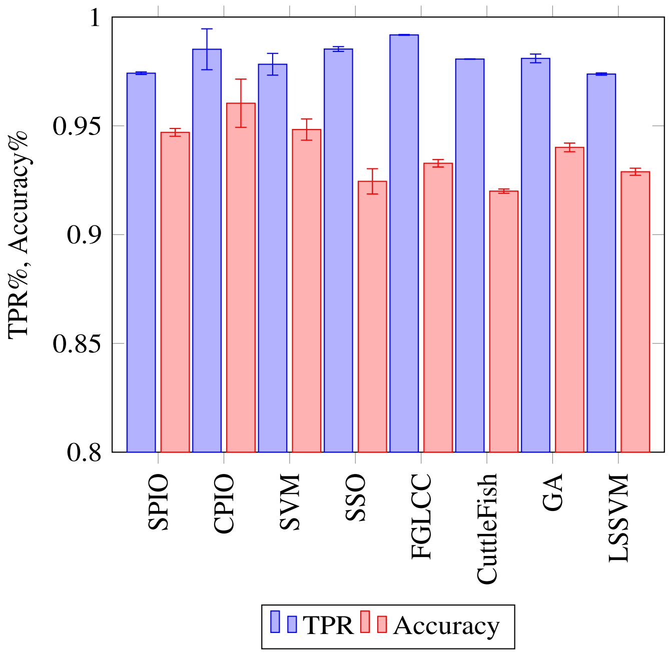

Fig. 7. TPR and Accuracy results of 8 different algorithms using *KDDCUP*99.

All examined algorithms were evaluated in terms of TPR, FPR, Fscore and accuracy. All reported results are an average of 30 runs. Table 6 presents all the related works used to compare with the proposed algorithms with the number of features nominated for the *KDDCUP*99 dataset. DT was used to evaluate each algorithm or technique used for feature selection by training the model using only the nominated features, and then the model was evaluated using the testing set. All the models trained on the same dataset with the same methodology to ensure fairness of comparison.

Fig. 6 shows the convergence curve for both binarize version of PIO for feature selection Sigmoid_PIO (SPIO) and Cosine_PIO (CPIO). The result shows that the proposed CPIO that used the cosine similarity to binarize the velocities of the solutions have a faster convergent than using the sigmoid function to binarize the velocities. Based on the results shown in Fig. 6, remember that the algorithms aim to minimize the fitness value of the solution.

It is clear from the Fig. 6 that the CPIO converges in exponential decay in the first 35 iterations and keeps enhance the quality of the solution, while the SPIO has a slower convergence than the CPIO and the quality of solution stopped enhancing at iteration 60. From the conducted results, the CPIO approved its efficiency compared to SPIO. The SPIO uses the traditional way of discretizing a continuous algorithm to apply it on a discrete problem such as feature selection. The proposed Cosine similarity method used to discretize the PIO is much faster to converge than the traditional way. The new pigeon entrance operation used by the CPIO helps the algorithm to keep enhancing the solution and skip the steadiness of the algorithm on a local optimal easily.

Fig. 7 illustrates the result of the TPR and accuracy for eight examined examined algorithms on *KDDCUP*99. Each bar represents the score and the standard deviation for 30 runs of each examined algorithm. As the figure shows that the proposed CPIO achieved the highest accuracy against all other examined algorithms. Also, the results show that the CPIO achieved a better result than SPIO in terms of both TPR and accuracy using the same number of iterations. The FGLCC algorithm achieved a high TPR compared to the examined algorithms, but it suffers from low accuracy. This due to the high false alarms rates that affect the accuracy. The model trained using the output of the Cuttlefish has the worst results in terms of accuracy. As observed from Fig. 7 algorithms with high TPR, and low accuracy did not take in consideration the false alarms rate in their fitness function.

Table 7 shows the comparison results for all examined algorithms by using the set of features produced by each algorithm that reported in Table 6 to train the DT classifier using 10% of the corrected *KDDCUP*99 training set. Table 7 presents the evaluation results of each model using *KDDCUP*99 corrected test set in terms of TPR, FPR, and Accuracy with a standard deviation of results. As the FGLCC achieved the highest TPR among all algorithms, but it suffers from high FPR. The proposed CPIO algorithm achieved the lowest FPR (False Alarms) compared to all examined algorithms which affect the results of the accuracy positively. Also, the proposed SPIO has nearly the same results achieved by the methodology that used SVM to select the features, but the model was trained using the features from SPIO is more stable than the one

Table 7
Comparison between several feature selection algorithms using *KDDCUP*99 corrected test set by Decision Tree (DT) in terms of TPR, FPR, Accuracy and NF.

Approach	TPR ± STDV	FPR ± STDV	Accuracy ± STDV	# of features
Proposed Sigmoid_PIO	0.974 ± (0.001)	0.097 ± (0.001)	0.947 ± (0.001)	10
Proposed Cosine_PIO	0.982 ± (0.009)	0.076 ± (0.0147)	0.960 ± (0.011)	7
Cuttlefish	0.980 ± (0.0001)	0.179 ± (0.002)	0.919 ± (0.001)	10
GA	0.981 ± (0.002)	0.128 ± (0.005)	0.940 ± (0.002)	10
SSO	0.985 ± (0.001)	0.174 ± (0.015)	0.924 ± (0.005)	6
FGLCC	0.991 ± (0.0002)	0.163 ± (0.004)	0.932 ± (0.001)	16
SVM	0.978 ± (0.005)	0.100 ± (0.009)	0.948 ± (0.004)	10
LSSVM	0.973 ± (0.001)	0.144 ± (0.004)	0.928 ± (0.001)	6

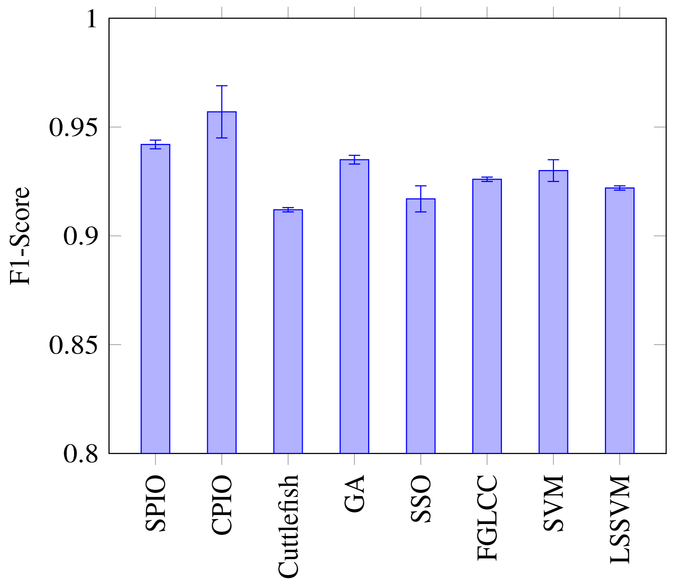

Fig. 8. F-score results for the 8 examined algorithms using *KDDCUP*99 dataset.

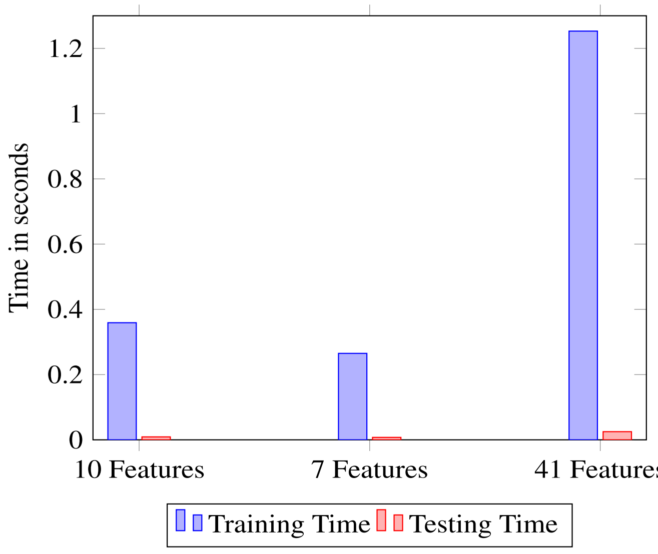

Fig. 9. Training and Testing time using Decision Tree on *KDDCUP*99 dataset.

uses the features from the SVM. In all measures, the proposed CPIO outperforms all other examined algorithms.

F-score measure summarizes and has a better indication than other measures, since it presents the results of precision and recall in a harmony way. Fig. 8 illustrates the results of F1-score for all examined algorithms. As the figure shows that the CPIO has the highest value of F1-score compared to others.

Another measure that affects the quality of a solution for feature selection algorithm is the number of selected features. The number of features affects the building and testing time of the model.

Fig. 9 illustrates the building and testing time for three cases: (1) using all features in the dataset (41 features), (2) using 10 features selected by the SPIO, and (3) using 7 features selected by the CPIO. The results show that the number of features affects the time needed to build and test the model.

The second dataset used to evaluate the proposed feature selection algorithm is NSL-KDD. As mentioned early in this paper, the NSL-KDD dataset is an improved version of the *KDDCUP*99 dataset and share the same features of it. Table 8 presents the selected set of features from NSL-KDD dataset by several feature selection algorithms including the proposed approaches. As it cleared from Table 8 that each feature selection algorithm has a different number of selected features.

Table 9 presents a comparison between several feature selection algorithms presented in Table 8. A DT classifier is trained using the set of selected features listed in Table 8 using NSL-KDD dataset and the results of these algorithms are compared in terms of TPR, FPR, accuracy and f-score. The results reported in Table 9 present an average of 30 runs for each algorithm, as the results show that the proposed Cosine_PIO has the highest TPR value, while GR algorithm achieved the lowest FPR. According to this, the accuracy and the f-score measure can be a better measure for comparing the examined algorithms. According to accuracy and f-score measures, the cosine_PIO has the best results with 0.883 and 0.882 for accuracy and f-score respectively. The sigmoid_PIO achieved the second-best results with 0.869 and 0.864, while IG came in the third place in terms of accuracy and f-score against the other examined algorithms presented in Table 9.

UNSW-NB15 is the third dataset used to evaluate the proposed PIO feature selection algorithm in this paper. Table 10 presents the selected set of features from UNSW-NB15 dataset by three feature selection algorithms with the proposed PIO algorithms. Each row contains the number of selected features with their index number and the feature selection used. Each index number of a selected feature can be mapped to feature name listed in Table 5.

Table 11 presents the average results of 30 runs for training and testing DT classifier on UNSW-NB15 and using the selected features by each algorithm presented in Table 10. The results presented in Table 11 show that the proposed CosinePIO has the lowest value of FPR against the other examined algorithms. The PSO have a good FPR compared to Cosine_PIO and better than other examined algorithms. In terms of accuracy and f-score the Cosine_PIO have the best results against the other examined algorithms. Also, the performance of the SigmoidPIO nearly the same of the Cosine_PIO.

Table 8
Selected set of features from NSL-KDD by several feature selection algorithms.

Reference	Technique	# of features	Selected set of features
(Shrivas & Dewangan, 2014)	GR	29	[1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 14, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39]
(Enache & Sgârciu, 2015)	BAT	18	[1, 2, 3, 8, 9, 13, 14, 18, 19, 20, 26, 28, 32, 33, 34, 38, 39, 40]
(Ambusaidi, He, Nanda, & Tan, 2016)	LSSVM	18	[3, 4, 5, 6, 12, 23, 25, 26, 28, 29, 30, 33, 34, 35, 36, 37, 38, 39]
(Moustafa & Slay, 2017)	Hybrid Association Rules	11	[2, 5, 6, 7,12, 16, 23, 28, 31, 36, 37]
(Aljawarneh, Aldwairi, & Yassein, 2018)	IG	8	[5, 3, 6, 4, 30, 29, 33, 34]
(Tama, Comuzzi, & Rhee, 2019)	PSO	37	[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41]
Proposed Approach	Sigmoid_PIO	18	[1, 3, 4, 5, 6, 8, 10, 11, 12, 13, 14, 15, 17, 18, 27, 32, 36, 39, 41]
Proposed Approach	Cosine_PIO	5	[2, 6, 10, 22, 27]

Table 9
Comparison between several feature selection algorithms using NSL-KDD dataset by Decision Tree in terms of TPR, FPR, Accuracy and F-score.

Approach	TPR ± STDV	FPR ± STDV	Accuracy ± STDV	F-score ± STDV
Proposed Sigmoid_PIO	0.817 ± (0.012)	0.064 ± (0.0008)	0.869 ± (0.006)	0.864 ± (0.006)
Proposed Cosine_PIO	0.866 ± (0.019)	0.088 ± (0.000)	0.883 ± (0.010)	0.882 ± (0.010)
GR	0.660 ± (0.022)	0.029 ± (0.000)	0.793 ± (0.010)	0.793 ± (0.010)
BAT	0.642 ± (0.008)	0.058 ± (0.001)	0.770 ± (0.004)	0.770 ± (0.004)
LSSVM	0.613 ± (0.003)	0.039 ± (0.003)	0.762 ± (0.002)	0.761 ± (0.002)
Hybrid Association Rules	0.665 ± (0.008)	0.035 ± (0.000)	0.796 ± (0.005)	0.795 ± (0.005)
IG	0.707 ± (0.013)	0.059 ± (0.003)	0.808 ± (0.007)	0.808 ± (0.007)
PSO	0.637 ± (0.012)	0.030 ± (0.000)	0.782 ± (0.008)	0.781 ± (0.008)

Table 10
Selected set of features from UNSW-NB15 by several feature selection algorithms.

Reference	Technique	# of features	Selected set of features
(Tama et al., 2019)	PSO	19	[6, 8, 9, 10, 11, 14, 19, 22, 24,26, 31, 33, 35, 36, 37, 42, 45, 46, 47]
(Kumar, Sinha, Das, Pandey, & Goswami, 2019)	Rule-Based	13	[5, 8, 9, 10, 13, 14, 32, 41, 42, 43, 45, 46, 47]
(Moustafa & Slay, 2017)	Hybrid Association Rules	11	[6, 10, 11, 19, 20, 27, 34, 37, 42, 44, 46]
Proposed Approach	Sigmoid_PIO	14	[3, 8, 9, 11, 12, 23, 26, 27, 28, 31, 38, 39, 40, 43]
Proposed Approach	Cosine_PIO	5	[3, 4, 8, 12, 29]

Table 11
Comparison between several feature selection algorithms using UNSW-NB15 dataset by Decision Tree.

Approach	TPR ± STDV	FPR ± STDV	Accuracy ± STDV	F-score ± STDV
Proposed Sigmoid_PIO	0.897 ± (0.0003)	0.052 ± (0.0004)	0.913 ± (0.0002)	0.904 ± (0.0002)
Proposed Cosine_PIO	0.894 ± (0.000)	0.034 ± (0.000)	0.917 ± (0.000)	0.909 ± (0.000)
PSO	0.863 ± (0.0004)	0.037 ± (0.0006)	0.895 ± (0.0003)	0.886 ± (0.0003)
Rule-Based	0.889 ± (0.005)	0.125 ± (0.005)	0.884 ± (0.003)	0.870 ± (0.004)
Hybrid Association Rules	0.721 ± (0.001)	0.057 ± (0.0003)	0.792 ± (0.0008)	0.784 ± (0.0008)

6. Conclusion

In this paper, a new feature selection algorithm based on pigeon inspired optimizer for IDS is proposed. The proposed PIO feature selection aimed to reduce the number of features needed to build robust IDS, while maintaining a high detection rate, accuracy with low false alarms.

The proposed PIO feature selection algorithm reduced the number of features of KDDCUPP99, NSL-KDD, and UNSW-NB15 datasets from 41 to only 7, 41 to 5, and 49 to 5 features for the three datasets respectively. It maintains a high TPR, accuracy and reduce the required time for building the model significantly.

Feature selection is a discrete optimization problem. A discretization process must be applied for a continuous swarm intelligent algorithm to solve such a problem. A new way of discretization a continuous algorithm that was based on the usage of cosine similarity is proposed and compared to the traditional way used by researchers. The proposed discretization process shows a faster convergence than the traditional way that used sigmoid function under the same number of iterations for PIO.

Author contributions

Use this form to specify the contribution of each author of your manuscript. A distinction is made between five types of contributions: Conceived and designed the analysis; Collected the data; Contributed data or analysis tools; Performed the analysis; Wrote the paper.

For each author of your manuscript, please indicate the types of contributions the author has made. An author may have made more than one type of contribution. Optionally, for each contribution type, you may specify the contribution of an author in more detail by providing a one-sentence statement in which the contribution is summarized. In the case of an author who contributed to performing the analysis, the author's contribution for instance could be specified in more detail as 'Performed the computer simulations', 'Performed the statistical analysis', or 'Performed the text mining analysis'.

If an author has made a contribution that is not covered by the five pre-defined contribution types, then please choose 'Other contribution' and provide a one-sentence statement summarizing the author's contribution.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Credit authorship contribution statement

Hadeel Alazzam: Conceptualization, Methodology, Software, Validation, Investigation, Writing - original draft, Writing - review & editing. **Ahmad Sharieh:** Supervision, Writing - review & editing. **Khair Eddin Sabri:** Supervision, Writing - review & editing.

Acknowledgments

All persons who have made substantial contributions to the work reported in the manuscript (e.g., technical help, writing and editing assistance, general support), but who do not meet the criteria for authorship, are named in the Acknowledgements and have given us their written permission to be named. If we have not included an Acknowledgements, then that indicates that we have not received substantial contributions from non-authors.

References

Acharya, N., & Singh, S. (2018). An iwd-based feature selection method for intrusion detection system. *Soft Computing, 22*(13), 4407–4416.
Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications, 60*, 19–31.
Al Shorman, A., Faris, H., & Aljarah, I. (2019). Unsupervised intelligent system based on one class support vector machine and grey wolf optimization for IoT botnet detection. Journal of Ambient Intelligence and Humanized Computing, 1–17.
Alazzam, H., Alsmady, A., & Shorman, A. A. (2019). Supervised detection ofiot botnet attacks. In *Proceedings of the Second International Conferenceon Data Science, E-Learning and Information Systems* (pp. 1–6).
Aljawarneh, S., Aldwairi, M., & Yassein, M. B. (2018). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science, 25*, 152–160.
Ambusaidi, M. A., He, X., Nanda, P., & Tan, Z. (2016). Building an intrusion detection system using a filter-based feature selection algorithm. IEEE Transactions on Computers, *65*(10), 2986–2998.
Ambusaidi, M. A., He, X., Tan, Z., Nanda, P., Lu, L. F., & Nagar, U. T. (2014). A novel feature selection approach for intrusion detection data classification. In 2014 IEEE 13th international conference on trust, security and privacy in computing and communications (pp. 82–89). IEEE.
Aslahi-Shahri, B., Rahmani, R., Chizari, M., Maralani, A., Eslami, M., Golkar, M. J., & Ebrahimi, A. (2016). A hybrid method consisting of ga and svm for intrusion detection system. Neural Computing and Applications, *27*(6), 1669–1676.
Asuncion, A., & Newman, D. (2007). Uci machine learning repository.
Basaran, D., Ntoutsi, E., & Zimek, A. (2017). Redundancies in data and their effect on the evaluation of recommendation systems: A case study on the amazon reviews datasets. In Proceedings of the 2017 SIAM international conference on data mining (pp. 390–398). SIAM.
Chen, K., Zhou, F.-Y., & Yuan, X.-F. (2019). Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. *Expert Systems with Applications, 128*, 140–156.
Chung, Y. Y., & Wahid, N. (2012). A hybrid network intrusion detection system using simplified swarm optimization (sso). *Applied Soft Computing, 12*(9), 3014–3022.
Deng, Y., & Duan, H. (2016). Control parameter design for automatic carrier landing system via pigeon-inspired optimization. *Nonlinear Dynamics, 85*(1), 97–106.
Duan, H., & Qiao, P. (2014). Pigeon-inspired optimization: A new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics, *7*(1), 24–37.
Eesa, A. S., Orman, Z., & Brifcani, A. M. A. (2015). A new feature selection model based on id3 and bees algorithm for intrusion detection system. *Turkish Journal of Electrical Engineering & Computer Sciences, 23*(2), 615–622.
Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing, 172*, 371–381.
Enache, A.-C., & Sgarciu, V. (2014). Enhanced intrusion detection system based on bat algorithm-support vector machine. In 2014 11th international conference on security and cryptography (SECRYPT) (pp. 1–6). IEEE.
Enache, A.-C., & Sgârciu, V. (2015). Anomaly intrusions detection based on support vector machines with an improved bat algorithm. In *2015 20th international conference on control systems and computer science* (pp. 317–321). IEEE.
Gupta, D., Joshi, P., Bhattacharjee, A., & Mundada, R. (2012). Ids alerts classification using knowledge-based evaluation. In 2012 fourth international conference on communication systems and networks (COMSNETS 2012) (pp. 1–8). IEEE.
Keshtgary, M., Rikhtegar, N., et al. (2018). Intrusion detection based on a novel hybrid learning approach. *Journal of AI and Data Mining, 6*(1), 157–162.
Kumar, G. (2014). Evaluation metrics for intrusion detection systems-a study. *Evaluation, 2*(11).
Kumar, V., Sinha, D., Das, A. K., Pandey, S. C., & Goswami, R. T. (2019). An integrated rule based intrusion detection system: analysis on unsw-nb15 data set and the real time online dataset. *Cluster Computing*, 1–22.
Lippmann, R. P., Graf, I., Wyschogrod, D., Webster, S. E., Weber, D. J., & Gorton, S. (1998). The 1998 darpa/afrl off-line intrusion detection evaluation. In First international workshop on recent advances in intrusion detection (RAID) (pp. 163–181).
Liu, H., & Motoda, H. (2012). *Feature selection for knowledge discovery and data mining*: 454. Springer Science & Business Media.
Maza, S., & Touahria, M. (2018). Feature selection algorithms in intrusion detection system: A survey.. *KSII Transactions on Internet & Information Systems, 12*(10).
Mohammadi, S., Mirvaziri, H., Ghazizadeh-Ahsaee, M., & Karimipour, H. (2019). Cyber intrusion detection by combined feature selection algorithm. Journal of Information Security and Applications, *44*, 80–88.
Moustafa, N., & Slay, J. (2015). Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In 2015 military communications and information systems conference (MILCIS) (pp. 1–6). IEEE.
Moustafa, N., & Slay, J. (2017). A hybrid feature selection for network intrusion detection systems: Central points. *arXiv preprint arXiv:1707.05505*.
Peddabachigari, S., Abraham, A., & Thomas, J. (2004). Intrusion detection systems using decision trees and support vector machines. *International Journal of Applied Science and Computations, USA, 11*(3), 118–134.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. Journal of Machine Learning Research, *12*(Oct), 2825–2830.
R. Al Shorman, A., Faris, H., Castillo, P., Merelo Guervs, J., & Al-Madi, N. (2018). The influence of input data standardization methods on the prediction accuracy of genetic programming generated classifiers. In *The 10th international joint conference on computational intelligence* (pp. 79–85). doi:10.5220/0006959000790085.
Revathi, S., & Malathi, A. (2013). A detailed analysis on nsl-kdd dataset using various machine learning techniques for intrusion detection. *International Journal of Engineering Research & Technology (IJERT), 2*(12), 1848–1853.
Ruggieri, S. (2002). Efficient c4. 5 [classification algorithm]. IEEE Transactions on Knowledge and Data Engineering, *14*(2), 438–444.
Sahu, S. K., Sarangi, S., & Jena, S. K. (2014). A detail analysis on intrusion detection datasets. In 2014 ieee international advance computing conference (IACC) (pp. 1348–1353). IEEE.
Scarfone, K., & Mell, P. (2012). Guide to intrusion detection and prevention systems (idps). *Technical Report*. National Institute of Standards and Technology.
Selvakumar, B., & Muneeswaran, K. (2019). Firefly algorithm based feature selection for network intrusion detection. *Computers & Security, 81*, 148–155.
Shewale, V. R., & Patil, H. D. (2016). Performance evaluation of attack detection algorithms using improved hybrid ids with online captured data. *International Journal of Computer Applications, 146*(8).
Shi, Y., et al. (2001). Particle swarm optimization: developments, applications and resources. In Proceedings of the 2001 congress on evolutionary computation (IEEE cat. no. 01th8546)*: 1* (pp. 81–86). IEEE.
Shrivas, A. K., & Dewangan, A. K. (2014). An ensemble model for classification of attacks with feature selection based on kdd99 and nsl-kdd data set. *International Journal of Computer Applications, 99*(15), 8–13.
Stolfo, J., Fan, W., Lee, W., Prodromidis, A., & Chan, P. K. (2000). Cost-based modeling and evaluation for data mining with application to fraud and intrusion detection. *Results from the JAM Project by Salvatore*, 1–15.
Sun, H., & Duan, H. (2014). Pid controller design based on prey-predator pigeon-inspired optimization algorithm. In 2014 IEEE international conference on mechatronics and automation (pp. 1416–1421). IEEE.
Tama, B. A., Comuzzi, M., & Rhee, K.-H. (2019). Tse-ids: A two-stage classifier ensemble for intelligent anomaly-based intrusion detection system. *IEEE Access, 7*, 94497–94507.
Tang, X., Dai, Y., & Xiang, Y. (2019). Feature selection based on feature interactions with application to text categorization. *Expert Systems with Applications, 120*, 207–216.
Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the kdd cup 99 data set. In 2009 IEEE symposium on computational intelligence for security and defense applications (pp. 1–6). IEEE.

Too, J., Abdullah, A. R., & Mohd Saad, N. (2019). Binary competitive swarm optimizer approaches for feature selection. *Computation, 7*(2), 31.

Varun, A., & Kumar, M. S. (2018). A comprehensive review of the pigeon-inspired optimization algorithm. *International Journal of Engineering & Technology, 7*(3.29), 758–761. doi:10.14419/ijet.v7i4.29.21654.

Yi, T.-H., Wen, K.-F., & Li, H.-N. (2016). A new swarm intelligent optimization algorithm: Pigeon colony algorithm (pca). *Smart Structures and Systems, 18*(3), 425–448.

Zaman, S., & Karray, F. (2009). Features selection for intrusion detection systems based on support vector machines. In 2009 6thIEEE consumer communications and networking conference (pp. 1–8). IEEE.

Zhang, B., & Duan, H. (2015). Three-dimensional path planning for uninhabited combat aerial vehicle based on predator-prey pigeon-inspired optimization in dynamic environment. IEEE/ACM Transactions on Computational Biology and Bioinformatics, *14*(1), 97–107.

Zhou, Y.-Y., & Cheng, G. (2019). An efficient network intrusion detection system based on feature selection and ensemble classifier. *arXiv preprint arXiv:1904.01352*.

Zorarpacı, E., & Özel, S. A. (2016). A hybrid approach of differential evolution and artificial bee colony for feature selection. *Expert Systems with Applications, 62*, 91–103.