A hybrid feature selection model based on butterfly optimization algorithm: COVID-19 as a case study

Ibrahim M. EL-Hasnony1 | Mohamed Elhoseny1,2 | Zahraa Tarek1

1Faculty of Computers and Information, Mansoura University, Mansoura, Egypt
2College of Computer Information Technology, American University in the Emirates, Dubai, UAE

Abstract

The need to evolve a novel feature selection (FS) approach was motivated by the persistence necessary for a robust FS system, the time-consuming exhaustive search in traditional methods, and the favourable swarming manner in various optimization techniques. Most of the datasets have a high dimension in many issues since all features are not crucial to the problem, which reduces the algorithm's accuracy and efficiency. This article presents a hybrid feature selection approach to solve the low precision and tardy convergence of the butterfly optimization algorithm (BOA). The proposed method is dependent on combining the algorithm of BOA and the particle swarm optimization (PSO) as a search methodology using a wrapper framework. BOA is started with a one-dimensional cubic map in the proposed approach, and a nonlinear parameter control technique is also implemented. To boost the basic BOA for global optimization, PSO algorithm is mixed with the butterfly optimization algorithm (BOAPSO). A 25 dataset evaluates the proposed BOAPSO to determine its efficiency with three metrics: classification precision, the selected features, and the computational time. A COVID-19 dataset has been used to evaluate the proposed approach. Compared to the previous approaches, the findings show the supremacy of BOAPSO for enhancing performance precision and minimizing the number of chosen features. Concerning the accuracy, the experimental outcomes demonstrate that the proposed model converges rapidly and performs better than with the PSO, BOA, and GWO with improvement percentages: 91.07%, 87.2%, 87.8%, 87.3%, respectively. Moreover, the proposed model's average selected features are 5.7 compared to the PSO, BOA, and GWO, with average features 22.5, 18.05, and 23.1, respectively.

Keywords

butterfly optimization algorithm, COVID-19, data classification, feature selection, particle swarm optimization

1 INTRODUCTION

There is a significant increase in data production due to the diversity of sources that generate it, which led to a big problem in producing useful information. There is a systematic need to store data, given the rising amount of data processed by applications in devices that make them internet access. Feature selection provides many functionalities like the decreased operational time by removing unnecessary and redundant attributes, improving performance precision of classification, and reducing the complexity of the classifier's organizational construction or the model definition. The performance of the clusters and the growth in the data collection dimensions have a significant impact on classification techniques. The data preprocessing stage's primary objective in the discovery information process is to make datasets available for data mining algorithms.
According to the FS system, the feature selection procedure consists of two main steps; firstly, search strategy and evaluation of sub-set quality secondly. The search strategy uses a method for selecting subgroups of features in the first stage. The next step consists of assessing the subset's quality as determined by the search strategy module using a classifier. On the other hand, three classes exist for feature selection strategies: wrapper, filter, and embedded-based techniques. The exhaustive search used in traditional methods for large data sets is not enough and takes a long time, so there are some limitations to look for the best variety of features. For example, if the feature size is d, it is hard to pick the required subset of features out of 2^d alternatives. The FS Wrapper method requires an internal classification to identify a more relevant subset of features, impacting its efficiency, particularly for massive datasets. Also, there are strategies for going backward and forward to incorporate or eliminate features that do not meet a broader range of specifications. According to these issues, the FS processes' efficiency is improved through metaheuristic algorithms (MA).

FS is a problem of optimization that aims to improve classification precision and reduce features number simultaneously. Metaheuristic methods are also promising alternatives to search in FS algorithms. These approaches are widely used to overcome various problems with optimization (Faris et al., 2020). There is thus great potential for similar output when a near-optimal subset of features is found. Recently, MA have been mainly shown to mimic the collective behaviour of organisms. In several fields of optimization, these algorithms have brought significant progress. MA could be the better choice because they can have the best outcomes in a reasonable time. Often, it could be a suitable alternative for reducing the time-consuming search constraints.

Conversely, many MA have a high locality standard, a lack of variety, and imbalances of exploration and exploitation. MA are divided into two groups, namely single-based and population-based metaheuristics (p). An evolutionary algorithm (EAs) is a type of population-based metaheuristics (p) algorithms. Feature selection begins with a thorough search over the feature subset within the EAs and discovers a specific evaluation criterion as the most attractive feature between the major possible subclasses. If the feature set includes n features, a feature subsets examination using an efficient feature selection procedure is needed to decide the best subset. Since evolutionary computing technology has a global search option, it is used to fix these problems with the best outcome and an alternative to classical search methods. The selection function has been widely used for the optimization of particle swarm (PSO), genetic algorithms (GA) and genetic programming (GP), and ant colony optimization (ACO).

MA is suitable for a wide range of implementations, including FS. Some classic methods have been used to efficiently fix the FS problem using GA, PSO, and differential evolution (DE). Moreover, modern MA algorithms such as competitive swarm optimizer (CSO), grasshopper optimization algorithm (GOA), gravitational search algorithm (GSA), and other algorithms have also been employed for FS. Since FS can be viewed as a problem of optimization, MA cannot manage all FS difficulties. This latter information is established according to no-free-lunch (NFL); consequently, the exploration of new alternative MA must continue (Youisi et al., 2020). Moreover, many researchers have also tried stochastic methodologies to solve feature selection problems such as PSO, GA (Kabir et al., 2011), (Bello et al., 2007), artificial bee colony (ABC) (Wang et al., 2010), and simulated annealing (SA) (Jensen & Shen, 2004), dragonfly algorithm (DA) (Tawhid & Dsouza, 2018) and grey wolf optimizer (GWO) (Emary et al., 2016) are the latest algorithms efficiently utilized to fix problems of feature selection. BOA, a recently evolved optimization algorithm, has influenced the researcher's enthusiasm because of its reliability, simplicity, and robustness of addressing real-world and engineering efficiency. To fix global optimization problems, BOA mimics food search and butterfly matching behaviour. In contrast to other optimization algorithms, BOA shows excellent efficiency (Arora & Singh, 2019). This metaheuristic based on the population can avoid local optima stagnation to some extent. It is also well able to converge towards the optimum. Arora and Singh (2017) utilized BOA to fix node locations in sensor networks wireless and compare the outputs with the firefly algorithm (FA) and PSO. Singh and Anand (2018) suggested a new algorithm for adaptive butterfly optimization to adjust the original BOA's sensory modality.

This paper's significant contributions are summarized into five folds. Firstly, a binary version of a new hybrid model (BOAPSO) is proposed for feature selection. The proposed hybrid model combines the BOA's functionality and the PSO for capabilities of exploration and exploitation, respectively. With exploration capability for the search area, the BOA has an excellent global convergence capacitance than other optimization algorithms. The PSO is power for BOA through preserving the search agent's experience. Secondly, the proposed BOAPSO is transformed into a binary version using the sigmoid transfer function that approved many enhancements according to the literature. The binary version of the proposed BOAPSO is utilized to select feature subsets using a wrapper framework with the classifier of K-nearest neighbour (KNN) for the evaluation process. For evaluating the proposed binary BOAPSO, the proposed model is applied to 25 standard feature subset selection datasets from the UCI machine learning repository and COVID-19 dataset. The proposed model achieves a better result according to three performance metrics: classification accuracy, selected feature set, and computational time. The proposed binary BOAPSO is compared to GWO, PSO, and BOA. The outcomes demonstrate the supremacy of the proposed model BOAPSO.

Thirdly, algorithms begin with the initial random population by investigating feature selection using MA. The techniques for initialization depend on randomness or compositionality. The Cubic map is used for this work because it is one of the popular maps in chaotic sequence generation in many implementations. The chaotic movement is characterized by randomness, regularity, and ergodicity. These features will prevent an optimum local problem from locking up the algorithm when solving feature optimization issues, sustaining demographic diversity, and enhancing universal search capabilities. Fourthly, a control approach for the nonlinear parameter is used to update the proposed model's position updating process. The linear parameters do not translate the nature of optimization in the convergence to the optimal solution. Lastly, the proposed
BOAPSO is compared to the recent works in most of the utilized datasets. Its supremacy is approved in terms of classification precision, chosen features, and computing time. The main contribution of this paper can be outlined as follows:

1. A new hybrid metaheuristic algorithm (BOAPSO) focused on the BOA and the PSO.
2. A binary version of the proposed BOAPSO for the FS process.
3. The cubic map is used for the initial population generation.
4. The nonlinear parameters utilized instead of the linear parameters in the native BOA.
5. The proposed binary BOAPSO evaluated by 25 datasets and approved its supremacy compared to the PSO, GWO, BOA, and some of the most recent related works.
6. The proposed BOAPSO applied to the COVID-19 dataset.

The remainder of the paper is arranged as follows: Section 2 introduces some of the previous works. Section 3 provides a background on the main concepts of the paper. Section 4 explains the proposed binary BOAPSO in detail, and Section 5 illustrates the outcomes and different relations. Section 5.4 presents some of the future search directions. Finally, the future work and conclusions are discussed in Section 6.

2 | RELATED WORKS

Because of its importance, many studies tried to enhance the feature selection process. Arora and Anand (2019) introduced paired alternatives of BOA to pick the optimum feature subset suitable for classification objects in wrapper procedure. The suggested binary algorithms were matched over 21 datasets available at the UCI repository using four high-performance optimization algorithms and with five approaches.

Tubishat et al. (2020) suggested dynamic butterfly optimization algorithm (DBOA) as an enhanced version for feature selection issues. Two significant changes have been made in the central BOA: introducing a local search algorithm based on mutation (LSAM) to prevent local optima problems and LSAM usage to increase the variety of BOA solutions. Twenty UCI repository benchmark datasets have been included. The experiments have shown that DBOA significantly outperforms comparable algorithms.

Rodrigues et al. (2020) suggested an individual, multi-and multi-objective paired alternatives of artificial butterfly optimization for feature selection. The trials were performed in eight common databases. The findings revealed that the binary single-objective is superior to the other meta-heuristic approaches, with a minimum number of chosen features. Regarding multi-and multi-objective function collection, both the suggested methods have done better than their single-objective meta-heuristic equivalents.

Abualigah et al. (2018) presented a strategy for selecting features using the PSO algorithm (FSPSOTC) to address choosing features by generating a new subgroup of informative features. Attempts were performed using six regular text datasets with a variety of features. The findings demonstrated that the suggested approach enhanced the text’s usefulness assemblage strategy by addressing a new subgroup of descriptive written features.

Yong Zhang et al. (2019) performed a feature selection process based on an unsupervised PSO algorithm, named a filter-based bare-bone particle optimization algorithm (FBPSO). Two techniques based on filter mode were suggested to improve the algorithm’s convergence; the first one was a space-reduction method focused on the average of reciprocal content, and the second was a search approach for redundancy of features using a local filter. Experimental findings on specific standard datasets have demonstrated the supremacy and efficacy of the presented FBPSO.

Qasim and Algamal (2018) suggested PSO along with the logistic regression method. Besides, a fitness function called Bayesian knowledge criterion (BIC) has been suggested. Experimental findings show the utility of the proposed approach to dramatically boost classification efficiency with minor features using various datasets. Furthermore, the outcomes confirmed that the recommended strategies had competitive efficiency relative to other known fitness functions.

Too et al. (2019) addressed the problem of feature selection for electromyography (EMG) signal categorization; a personal best mode for binary particle swarm optimization (PBPSO) was suggested for solving this issue. Sadeghian et al. (2021) suggested binary butterfly optimization algorithm for information gain (IG-bBOA) to solve binary butterfly optimization algorithm in the form of an S (S-bBOA) constraints. The outcomes were based on six routine UCI registry datasets. The results demonstrated the efficacy of the suggested approach in enhancing the precision of classification and choosing the best optimum features of the subset with minimal features in most situations.

Li et al. (2019) established and developed BOA by incorporating the cross-entropy (CE) approach into the initial algorithm. The suggested solution’s efficiency depended on 19 common benchmark assessment mechanisms and three widespread engineering design difficulties. The test function results referred to the supremacy of the proposed algorithm, as it could deliver promising results for local optima avoidance, enhanced discovery, and exploitation reduction.

Abualigah and Khader (2017) suggested a PSO algorithm with genetic operators for an FS problem. The k-means clustering approach was used to determine the utility of the function subsets obtained. The results were obtained by analyzing eight standard text datasets with varying features.
Ibrahim et al. (2019) suggested a crossbred optimization approach for the feature selection issue, coupled with a slap swarm algorithm and a particle swarm optimization method (SSAPSO). To test the efficacy of the proposed algorithm, it was examined across two experimental ranges; firstly, it was compared with other related methods. Secondly, SSAPSO was utilized to evaluate the optimal features set using separate UCI benchmark datasets.

Tawhid and Dsouza (2018) suggested a hybrid binary dragonfly and enhanced particle swarm optimization (HBDESPO) for manipulating the feature selection issue.

Indicating to the theorem of NFL (Wolpert & Macready, 1997), there is no particular algorithm suitable for all forms of FS problems. The algorithm’s success in solving a specific feature selection problem does not ensure comparable results when applied to other FS issues. From this view, there are several possibilities for developing more efficient FS systems by introducing novel algorithms or developing derivatives of existing systems.

3 | PRELIMINARIES

The principles used in this article include the feature selection procedure, particle swarm optimization algorithm, the butterfly optimization algorithm, and a comparison between the BOA and different MA, which are covered in detail in the following subsections.

3.1 | Feature selection

Among the most common methods proposed in machine learning is feature selection. Feature extraction aims to eliminate redundant features and choose the most appropriate features from among the main features to enhance learning algorithms’ effectiveness. The two most important criteria are the selection and construction features in machine learning (ML). The two variables are generally very time-consuming and complex tasks, as the characteristics need to be manually designed. Attributes are aggregated, merged, or separated to generate features from raw data (Moslehi & Haeri, 2020).

It is typically challenging to perform a comprehensive search to locate the most features in computing costs. The reduction of features has therefore been a significant problem in machine learning pattern recognition. This technique enjoys excellent attention in several applications, including regression and categorization; since there are typically many features in these applications, most of them lead to decreased performance precision or inefficient. Deleting these features reduces the computational complexity, in addition to increasing accuracy (Jović et al., 2015).

Feature selection techniques aim to get the most useful subset of the N feature and $2^N$ subsets. In both approaches, a subset is chosen as an answer such that the evaluation mechanism can be refined depending on the application and form of description. While each system attempts to identify the most critical features in terms of the extent of potential answers, seeking an optimal solution is challenging and relatively expensive in large and medium-sized datasets. Three key types, namely a wrapper, filter, and embedded versions, can be used to define the method of selecting features, as seen in Figure 1.

![Figure 1](image.png)

**Figure 1** The main categories of feature selection model. (a) Filter approach. (b) Wrapper approach. (c) Embedded approach.
3.1.1 | Filter approaches

To solve feature selection, the statistical methodology is employed to feature compilation in the filter process (Moradi & Gholampour, 2016). Filter modes assess and pick the features' significance utilizing a rating system that eliminates unnecessary features. Filter approaches have been demonstrated to be rapid, scalable, simple computationally, and independent. Similar methods are classified into two types: multivariate filter methods and univariate filter methods.

3.1.2 | Wrapper approaches

Wrapper approaches are based on a particular machine learning algorithm when selecting the feature. The chosen feature subset is used to train the learner directly in the screening process and determine the advantages of the feature subset depending on the learner test collection results. The approach is not as practical as the filter approaches, but the chosen subset size of features is comparatively slight. In this approach, the generation algorithm (GA) develops each new feature subset, and the search process determines this output. In general, the wrapper method is more efficient than the filter approach, but it is more computationally complicated (Tang et al., 2014).

3.1.3 | Embedded methods

Differences from most ways of choosing features are the manner of learning interaction and feature selection. The methods of filtering do not integrate learning. Wrapper approaches use machine learning to calculate the consistency of feature subsets without incorporating awareness of the essential essence of the regression or classification method and can thus be used with any learning machine. Unlike filtering and wrapper, embedded approaches do not isolate learning from the feature's selection aspect—the class structure of functions represents a fundamental role (Lal et al., 2006).

3.2 | Particle swarm optimization

The concept of the particle swarm optimization (PSO) algorithm is established by the behaviour of social foraging of certain species, such as the schooling behaviour of the fish and the flocking behaviour of birds. The PSO algorithm is made up of particles; each particle has its velocity and position. The objective function will be examined after each position updates. Particle clusters meet over time surrounding single or multiple optima with a mixture of known locations in the search space (Brownlee, 2011).

PSO is a statistical approach that improves the problem by recursively trying to enhance the nominee solution for a given quality metric (Golbon-Haghighi et al., 2018). PSO has many likenesses with evolutionary programming methods such as genetic algorithms. PSO's main power is its fast convergence, distinguishing it from specific global optimization algorithms such as simulated ringing, genetic algorithms, and other optimization methods (Umarani & Selvi, 2010).

The simplest version of the PSO algorithm operates by providing a population or a swarm of nominee solutions (named particles). PSO improves the problems by generating a population of particles and carrying them around in the quest space using simple mathematical formulas to calculate the particle's location and velocity. Its local best-known position guides every particle's movement. It is also driven to the most prominent places in the search-space, modified as better positions get from other particles. This is supposed to move the population towards the correct solutions to the assigned problem (Yudong Zhang et al., 2015).

Particle motions rely on the best local and global in each iteration; each particle has its best local (the best location obtained by that particle) and the best global (best position from all the local best) (Mathiyalagan et al., 2010). Parameters for optimization methods are presented in Table 1 (Al-Khafaji & Abdulla Al-Kabragyi, 2011), (Pereira, 2011).

An n-dimensional vector, \( \mathbf{X}_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \), represents the location of \( \mathbf{x}_i \) particle in the whole population. Also, the n-dimensional vector \( \mathbf{V}_i = (v_{i1}, v_{i2}, \ldots, v_{in}) \) represents the velocity of the specified particle. And, \( \mathbf{P}_i = (p_{i1}, p_{i2}, \ldots, p_{in}) \) denotes the best-visited place previously of the \( \mathbf{x}_i \) particle. ‘g’ is used as the best particle index overall population. Equation (1) is used to update the velocity of the \( \mathbf{x}_i \) particle:

\[
v_{i}^{t+1} = v_{i}^{t} + c_1 r_1 (p_{best}^{t} - x_i^{t}) + c_2 r_2 (g_{best}^{t} - x_i^{t})
\]

And the location of this particle is calculated using Equation (2):
Table 1: Parameters of optimization techniques

| Parameter | Denotation |
|-----------|------------|
| $X^k$     | Current position of particle $i$ at iteration $k$ |
| $X^{k+1}$| Position of the particle $i$ at iteration $k + 1$ |
| $V^k$     | Velocity of particle $i$ at iteration $k$ |
| $V^{k+1}$| Velocity of particle $i$ at iteration $k + 1$ |
| $W$       | Inertia weight between 0.9 to 0.1 |
| $c_j$     | Positive acceleration coefficients; $j = 1, 2$ |
| rand$_i$  | Random number between 0 and 1; $i = 1, 2$ |
| pbest$_i$ | Best position of particle $i$ |
| gbest     | Position of best particle in a population |

Algorithm 1

Particle swarm optimization (PSO)

Input: population size ($S$), particle position ($X$), inertia weight ($W$), and learning parameters [$c_1, c_2$], solution dimension ($d$), and maximum number of iteration ($T_{max}$).

Output: optimum solution ($g_{best}$)

1. Start
2. while $t < T_{max}$
3. Evaluate each particle fitness
4. For $i = 1: S$
5. Find $p_{best}$ (the better value for each particle from start)
6. Find $g_{best}$ (the overall best value)
7. For $j = 1: d$
8. Velocity update using Equations (1) and (3)
9. Positions update using Equation (2)
10. End for
11. Adjust the inertia weight ($W$)
12. End for
13. End
14. End

$x^{t+1}_i = x^t_i + v^{t+1}_i$  \hspace{1cm} (2)

as $i = 1: 2\ldots S$; and $S$ is the swarm’s size; $c_1$ and $c_2$ are factors of constant cognitive and social scaling. The inertia weight ($w$) has been reported previously, so Equation (3) for the velocity update becomes:

$v^{t+1}_i = wv^t_i + c_1r_1(p_{best}^t_i - x^t_i) + c_2r_2(g_{best}^t_i - x^t_i)$  \hspace{1cm} (3)

The PSO algorithm has been considered in this paper, so it will be discussed according (Sarangi & Thankchan, 2012). The pseudo-code of the PSO is given in Algorithm 1.

There was no inertia weight in the initial PSO, but the inertia weight was added to boost performance by researchers. Then efficiency is attempted to improve by carrying out various initialization methods. Researchers are still working on the global best particle to escape the nearby minima. For this reason, the different mutation operators are added to boost the efficiency of the PSO (Imran et al., 2013). The flowchart of PSO is presented in Figure 2.
3.3 Butterfly optimization algorithm

3.3.1 Butterfly biological behaviour

Butterflies are classified within the Lepidoptera family in the Animal Kingdom’s Linnaean classification scheme. Around the globe, there are over 18,000 different species of butterflies. Their senses are responsible for their long life span of millions of years (Saccheri et al., 1998). Butterflies have some senses like sight, smell, touch, taste, and hearing used to locate food and mating partner. Other benefits from these senses include hiding from predators, transporting from one location to another, and laying eggs in suitable places. The smell is the most significant of all these senses that allow butterflies to locate food, often from a long distance, usually nectar. Figure 3 displays some images of the butterflies.

Nature-inspired MA have drawn a great deal of interest from numerous researchers in the past (Yang, 2010). Butterfly optimization algorithm (BOA) is a significant subcategory of MA from nature’s inspiration. Butterflies’ food searching activity fundamentally influences BOA, and these insects are used as search factors for optimization in BOA (Arora et al., 2018).
3.3.2 | Movement of butterflies

BOA is the population-based, biologically inspired algorithm suggested by S. Arora et al. and is a standard optimization algorithm. In 2018, the BOA imitated butterfly food and social behaviour. In BOA, butterflies are believed to have a particular energy/intensity scent/fragrance. This fragrance has to do with the butterfly’s fitness, measured using the problem’s objective function. This means that if a butterfly moves in the quest space from one location to another, its fitness will update. There are various butterflies in the neighbourhood that can feel with the fragrance produced by the butterfly. If a butterfly senses the fragrance of the best butterfly in the quest space, it will work its way towards it, and this stage is referred to as the BOA global search stage.

In the second example, if a butterfly cannot identify another butterfly’s scent in the search field, it will take random steps, referred to as the local search stage. The scent in BOA is formed as a function of the physical strength of the stimulus, as seen in Equation (4):

$$p_{fi} = c \times i^a$$

where $p_{fi}$ is the relative intensity of the scent, that is, how strongly other butterflies in the region perceive the fragrances of $i$th butterfly, the sensory modality is denoted by $c$, the stimulus intensity is $I$ and $a$ is the strength exponent that varies with modality, and accounts for the absorption degree. In BOA, an artificially positioned butterfly can be modified using the optimization procedure, as shown in Equation (5):

$$x_{iT} = x_{iT} + F_{iT}$$

where $x_{iT}$ represents the solution vector for $i$th butterfly in iteration sequence $t$ and $F_{iT}$ describes scent, which $x_{iT}$th butterfly uses to upgrade its location throughout all iterations. In addition, the algorithm includes two key steps: local and global search. During the stage of global search, the butterfly moves closer to the best solution $g^*$, as illustrated in Equation (6):

$$F_{IT} = (r^2 \times g^* - x_{iT}) \times p_{fi}$$

where $x_{iT}$ and $x_{kT}$ come from the space solutions $j_{on}$ and $k$th butterflies. If $x_{iT}$ and $x_{kT}$ are in the same swarm and $r$ is a random number in range between 0, 1. So Equation (7) is a haphazard local stroll. BOA employs a transfer probability $p$ to transition from global search to local search. The pseudo-code of the BOA is given in Algorithm 2.

4 | THE PROPOSED HYBRID MODEL FOR FEATURE SELECTION

This section provides the steps and the sequence of the proposed model in detail—the block diagram for arranging the proposed model processes displayed in Figure 4. As seen in Figure 4, the model first initialized with a set of parameters and random solutions using the cubic map sequence. The next step involved the objective function evaluation of the population initialized by the cubic map. Finally, the optimization process or the position updating is performed for every candidate solution using the hybrid between the butterfly optimization algorithm and the particle swarm optimization algorithm (BBOAPSO). A novel hybrid HPSOBOA is proposed in this section to combine the advantages of the three improvement strategies presented in this paper, which are the cubic map for the initial population, nonlinear parameter control strategy of power exponent $a$, PSO algorithm, and BOA algorithm. These steps are provided in algorithm three and discussed in detail in the following subsections.

4.1 | Cubic map initialization

The first step in the proposed model is the method by which $n$ butterflies, or search agents, are initialized in random form. Each search agent is a workable alternative, and length $D$ equals the number of features in the initial dataset and represents a desirable solution. The potential solution of the dataset, including $d$ features, an example is shown in Figure 5. To this end, $M$ records and $D$ features are first loaded with data.
Therefore, the aim is to identify the feature chosen from the D number of features to reduce the problem dimension provided that the main concern is not damaged. It is, therefore, essential to decide which one of these D features can maximize the accuracy of classification.

The problems with selecting the classification features are summarized by choosing a specific subset of the features that maximize the classification accuracy. Initially, binary values (0 and 1) were set in each solution. We must therefore define the relevant characteristics (one value) and ignore others (zero). There are several random initializing methods such as distributed sampling (DS), chaotic maps, etc. Recently, chaotic arrays were used instead of random number sequences in many applications. The chaotic movement characterized by regularity, randomness, and ergodicity. These features will avoid a local optimization problem-free algorithm for addressing function optimization issues, maintaining population diversity, and improving global search functionalities. Chaotic maps have many forms, like logistic maps, tent maps, circle map, cubic map, gauss map, ICMIC map, and sinusoidal iterator (Lu et al., 2014).

In nonlinear systems, chaos is a relatively common phenomenon. Cubic map is one of the most widely used maps for chaotic sequences generated in several applications. This map is defined formally by Equation (8) (Rogers & Whitley, 1983):

\[ x_{k+1} = \rho x_k (1 - x_k^2), \quad x_k \in (0, 1). \]  

where the \( \rho \) denotes to control parameter, in Equation (8), the cubic map sequence is in \( (0, 1) \), when \( \rho = 2.595 \), the chaotic variable \( x_{k+1} \) produced at this time has better ergodicity. A graphical presentation of the cubic map is in Figure 6.

4.2 Updating the positions

The proposed BPSOBOA binary feature selection model, a combination of separate PSO and BOA, is discussed in this section. The most significant gap between PSO and BOA is the generation of new individuals. The PSO algorithm’s disadvantage is that it covers a small space to resolve
**FIGURE 4**  The block diagram for the proposed feature selection model

**FIGURE 5**  Problem-making method for the proposed model
problems with high-dimensional optimization. To consolidate the two algorithms’ benefits, both algorithms’ functionality is combined, and the algorithms are not used one after the other. In other words, the method used to produce the results of these two algorithms is heterogeneous. The following equations establish the way to generate the following position values:

\[ x_{i+1} = x_i + (r^2 \times g_{\text{best}} - x_i) \times f_i \]  \hspace{1cm} (9)

where The fragrance \((f_i)\) can be formulated as follows:

\[ f_i = c \times t^a \]  \hspace{1cm} (10)

where \(c\) represents the sensory modality, \(f_i\) represents the perceived magnitude of fragrance; \(a\) means the power exponent based on the degree of fragrance absorption, \(I\) define the stimulus intensity, and \(a\) represents the power exponent based on the degree of fragrance absorption.

Algorithm 3

The proposed BBOAPSO feature selection model

\textbf{Input} — agent position \(X\), total number of iteration \(T_{\text{max}}\), population size (\(N\)), swing factor \(C\), feature dimension \(d\), control parameter \(a\), switch probability \(p\), sensory modality \(c\), the initial value of power exponent \(a\), and learning factors \(c1,c2\).

\textbf{Output} — Optimal solution

1. Begin
2. For \(i = 1: N\)
3. For \(j = 1: d\)
4. Generate the cubic chaotic sequence according to Equation (8)
5. End for
6. End for
7. //use cubic to initialize the population
8. For \(t = 1: T_{\text{max}}\)
9. For \(i = 1: N\)
10. For \(j = 1: d\)
11. Update the fragrance of current search agent by Equation (10)
12. End for
13. End for
14. Find the best \(f\)
15. For \(i = 1: N\)
16. For \(j = 1: d\)
17. Set a random number \(r\) in \([0,1]\)
18. If \(r < p\), then
19. Move towards best position by Equation (9)
20. Else
21. Update the velocity using Equation (13)
22. Update the position by Equation (14)
23. End if
24. End for
25. End for
26. Update \(a\) according to Equation (11)
27. Update \(c\) according to Equation (12)
28. Update \(W\) according to Equation (15)
29. End for
30. End
We can see the role of the power exponent \(a\) in the ability of BOA is essential to find the best optimization. If \(a = 1\) indicates that no scent is absorbed — that is, other butterflies will perceive the scent issued by a particular butterfly — thus narrowing the search range and enhancing local algorithm exploration. If \(a = 0\) is not perceivable to any Butterfly, the fragrance will expand to include a search range, that is, improve the algorithm's global exploratory capability. However, if \(a = 0.1\), it cannot effectively balance the basic BOA search capabilities and take a fixed value. Consequently, we propose an Equation (11) strategy for nonlinear parameter control (M. Zhang et al., 2020):

\[
a(t) = a_1 - (a_t - a_1) \times \sin \left( \frac{\pi}{\mu} \left( \frac{t}{T_{\text{max}}} \right)^2 \right)
\]

where \(a_1\) and \(a_t\) represent the initial value and final value of the parameter, \(\mu\) is the tuning parameter, and \(T_{\text{max}}\) represents the maximum number of iterations.

A value of \(c\) sensory morphology in the range [0,1] can be theoretically taken. Its value is however, dependent upon its specificity in the iterative BOA process as the optimization problem. The sensory modality \(c\) can be formulated as Equation (12) in the optimal research phase of the algorithm:

\[
c_{t+1} = c_t + \left( \frac{0.025}{(c_t \times T_{\text{max}})} \right)
\]

where \(T_{\text{max}}\) is the maximum number of iterations of the algorithm, and the initial value of parameter \(c\) is set to 0.01.

Global and local food search can take place, as well as a butterfly matching partner in nature. A switch probability \(p\) is therefore set to switch between global search and local intensive search. The BOA generates a [0,1] number on a random basis, compared to a \(p\)-value to determine whether a global search is to be performed or a local search conducted. If the random number is less than \(p\), the position will be updated according to Equation (9), otherwise, the position updated according to Equations (13) and (14).

\[
v_{i+1} = wv_t + c_1r_c(p_{\text{best},t} - x_t^t) + c_2r_r(g_{\text{best},t} - x_t^t)
\]

\[
x_{i+1}^t = x_t^t + v_{i+1}^t
\]

where \(v_t\) and \(v_{i+1}\) represent the velocity of the \(i\)th particle at iteration number \(t\) and \((t + 1)\). Usually, \(c_1 = c_2 = 2\), \(r_c\), and \(r_r\) are the random numbers in (0, 1). The \(w\) can be calculated as Equation (15):

\[
w(t) = \frac{w_{\text{max}} - W_{\text{min}}}{T_{\text{max}}} \times T_t
\]

where \(w_{\text{max}} = 0.9\), and \(w_{\text{min}} = 0.2\), and \(T_{\text{max}}\) represents the maximum number of iterations. Max and Min are the maximum and minimum values in the continuous feature vector, respectively.
4.3 | Objective function evaluation

Feature selection can also be seen as a problem of multi-target optimization. The best solution in BOAPSO includes the minimum number of characteristics with the most significant classification accuracy. The fitness function, therefore, has been formulated as Equation (16) (Abdel-Basset et al., 2020). Based on this, the fitness function of the solutions assessment is designed to balance the objectives as follows:

$$fitness = \alpha R_D(D) + \beta \frac{|S|}{|D|}$$

(16)

where $|S|$ represents the cardinality of the selected features set, $R_D(D)$ is the error rate of the classifier, $|D|$ represents the total features cardinality of the original dataset. $\alpha$ and $\beta$ are measurement parameters that reflect the value of the accurate classification and the selected features set, $\alpha \in [0,1]$ and $\beta = 1-\alpha$, and these values have been determined based on the evaluation function.

The Euclidean distance (Gou et al., 2019), used in KNN, is thus evaluated as in Equation (17) to evaluate the K neighbours adjacent to this sample:

$$EUC_d(P, Q) = \sqrt{\sum_{i=1}^{d} (Q_i - P_i)^2}$$

(17)

where $Q_i$ and $P_i$ are represented for a given record in the dataset for specific attributes, and $i$ is a variable from 1 to $d$. A common method is to save part for the validation dataset, and the rest can be used for the classification training. However, if we do, we can probably confront the overfitting problem, when the accuracy of a particular classifier is more than the test data for learning. Cross-validation is a popular way to reduce the overfitting problem. K -fold cross-validation with $K = 10$ is implemented in this paper. It is assumed that the samples should be divorced in $K$ folds or partitions of the roughly same size. The classifier is trained in $K$-1 and then tested for predicting each sample to which class label the remainder of the partition belongs. The proportion is then evaluated for the inappropriate class mark estimate known as the percentage error rate of classification. The results of various data rounds are statistically accurate on average (Wong & Yeh, 2019).

4.4 | Binary transformation

The generated values of the search agent positions are continuous. Since there are conflicts with the standard binary format for selecting features, it is not directly applicable. The best features are chosen to improve a specific classification algorithm's performance and accuracy, according to the feature selection problem with values (0 or 1). By transforming values from continuous to binary, the result/calculated search space will be changed. As seen in Figure 7, the sigmoidal function is an S-shaped function example (Abdel-Basset et al., 2020).

Any continuous value can be translated into binary by the sigmoid function using the following Equations (18) and (19):

$$x_{binary} = \begin{cases} 
0 & \text{if } R < x_i \\
1 & \text{if } R \geq x_i
\end{cases}$$

(19)

where $x_i$ in the S-shaped search agent is a continuous value (feature), $i = 1, \ldots, d$, and $x_{binary}$ value can be 0 or 1 by random number $R \in [0,1]$ value compared to $x_i$.

5 | EXPERIMENTAL RESULTS

To validate the proposed model's performance for COVID-19 detection, a set of experimental series are performed. The first experimental series’ main aim is to evaluate the performance of the proposed BOAPSO model through a set of 25 UCI datasets. Meanwhile, the second practical series aims to test the developed COVID-19 detection method’s applicability using the COVID-19 dataset.
5.1 | Experimental series 1: Feature selection using UCI data-sets

This section offers a broad, empirical analysis of the behaviour based on many improvements to the BOAPSO optimization algorithm. In this paper, 25 data sets are used. Several experiments are used to validate the performance of the proposed BOAPSO for features selection. The whole experiments were conducted in the Windows 10 Pro 64-bit operating system; processor Intel(R) 16 GB of RAM Core (TM) i7-8550U CPU@ 1.80GHZ 1.99 GHz. All algorithms are applied using MATLAB.
### Table 3 Parameters setting

| Parameters                              | Value |
|-----------------------------------------|-------|
| Iterations                             | 20    |
| Independent runs                       | 20    |
| K-folds cross-validation                | 10    |
| K-neighbours                           | 5     |
| Search agents (n)                      | 5     |
| Tuning parameter (μ)                    | 2     |
| $C_1 = C_2$                             | 2     |
| The initial value of sensory morphology coefficient $τ_c$ | 0.01  |
| $W_{max}$                              | 0.9   |
| $W_{min}$                              | 0.2   |
| $A_f$                                   | 0.3   |
| $A$                                     | 0.9   |
| $B$                                     | 0.1   |
| $A_s$                                   | 0.1   |
| $r$                                     | 0.1   |

### Table 4 The proposed BOAPSO results

| Dataset   | Original accuracy | Accuracy (proposed) | Computational time | All features | Selected features (proposed) |
|-----------|-------------------|---------------------|--------------------|--------------|-------------------------------|
| DS_1      | 88.57             | 97.1                | 16.5               | 299          | 13.8                          |
| DS_2      | 96                | 96.9                | 7.3                | 9            | 2.3                           |
| DS_3      | 61.6              | 70.6                | 5.6                | 19           | 4.1                           |
| DS_4      | 87.1              | 88.9                | 6.1                | 23           | 2.1                           |
| DS_5      | 78.9              | 91.9                | 5.11               | 22           | 2.9                           |
| DS_6      | 92.5              | 97.3                | 5.9                | 30           | 4.1                           |
| DS_7      | 89                | 96.4                | 4.3                | 16           | 5.7                           |
| DS_8      | 90.48             | 95.2                | 6.5                | 20           | 2.4                           |
| DS_9      | 85.1              | 91.4                | 6.2                | 34           | 2                             |
| DS_10     | 81.4              | 82.9                | 8.3                | 21           | 3.9                           |
| DS_11     | 95.4              | 96.7                | 12.2               | 10           | 2.1                           |
| DS_12     | 92.5              | 93.9                | 4.5                | 21           | 2                             |
| DS_13     | 68.8              | 82.5                | 6.2                | 90           | 7.8                           |
| DS_14     | 95                | 95.8                | 5.1                | 19           | 3.9                           |
| DS_15     | 71.5              | 92.8                | 5.8                | 61           | 9.1                           |
| DS_16     | 79                | 85.3                | 6.2                | 22           | 1.9                           |
| DS_17     | 84                | 95                  | 7.9                | 9            | 2.9                           |
| DS_18     | 72                | 94.5                | 6.2                | 18           | 4.9                           |
| DS_19     | 92.3              | 98.2                | 6.9                | 13           | 3.1                           |
| DS_20     | 95.2              | 98.5                | 12.3               | 40           | 12                            |
| DS_21     | 77                | 79.3                | 4.5                | 9            | 4.2                           |
| DS_22     | 92.5              | 96.8                | 3.6                | 16           | 4.7                           |
| DS_23     | 82.4              | 82                  | 3.4                | 18           | 5.7                           |
| DS_24     | 86.2              | 94.7                | 5.8                | 13           | 5.8                           |
| DS_25     | 89.5              | 98.2                | 6.1                | 265          | 30.7                          |
5.1.1 Dataset description

The proposed algorithm (BOAPSO) was implemented on 25 datasets received from the UCI repository to evaluate the potency of this approach (Dheeru & Taniskidou, 2017). Table 2 introduced these datasets. The data set contains various variables, classes, and instances to prepare an overall and broad survey of the proposed and used approaches to selecting features. The primary reason why these data sets are chosen is that they contain several attributes and instances, which are a variety of problems where the binary proposed approach is tested.

Moreover, to assess the performance of the proposed BOAPSO in the high dimensional search areas, a set of high dimensional data sets is also selected. Each data set is cross-validated for evaluation purposes. The dataset is divided into $K - 1$ folds for cross-validation training, and the rest of the folding is used for testing. This is repeated for $M$ times. Thus, there are $K \times M$ times for each data set evaluated for each optimization algorithm. Data are distributed into parts of equal size for training, testing, and validation. The training portion is devoted to the classifier training during the optimization process, while the validation portion is used to evaluate the classifier performance during the optimization period. The test fraction is used to assess the selected feature of the trained classifier.

**FIGURE 8** Comparison between the proposed BOAPSO classification accuracy and the original dataset classification accuracy

**FIGURE 9** A comparison between the proposed BOAPSO features set and the original dataset features

5.1.1 Dataset description

The proposed algorithm (BOAPSO) was implemented on 25 datasets received from the UCI repository to evaluate the potency of this approach (Dheeru & Taniskidou, 2017). Table 2 introduced these datasets.

The dataset contains various variables, classes, and instances to prepare an overall and broad survey of the proposed and used approaches to selecting features. The primary reason why these data sets are chosen is that they contain several attributes and instances, which are a variety of problems where the binary proposed approach is tested.

Moreover, to assess the performance of the proposed BOAPSO in the high dimensional search areas, a set of high dimensional data sets is also selected. Each data set is cross-validated for evaluation purposes. The dataset is divided into $K - 1$ folds for cross-validation training, and the rest of the folding is used for testing. This is repeated for $M$ times. Thus, there are $K \times M$ times for each data set evaluated for each optimization algorithm. Data are distributed into parts of equal size for training, testing, and validation. The training portion is devoted to the classifier training during the optimization process, while the validation portion is used to evaluate the classifier performance during the optimization period. The test fraction is used to assess the selected feature of the trained classifier.
5.1.2 Evaluation criteria

To evaluate the proposed BOAPSO, three measures are utilized as follows (Arora & Anand, 2019):

1. **Classification accuracy**: Is an indicator that explains how the classification is accurate given the set of features chosen; that is, the correct classification number of features and the precision of classification in this study is determined as Equation (20):

   \[
   \text{Average performance} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N} \sum_{i=1}^{N} \text{match}(C_i, L_i) \quad (20)
   \]

   where \( M \) is the number of times the optimization algorithm is performed, \( N \) indicates the number of test set points, \( C_i \) indicates the output class label of a unique Data point \( i \), \( L_i \) is a reference class label for \( i \) and corresponds to the comparison function, which gives output 1 if two labels are the same and output 0 otherwise.

2. **Average selection size**: Represents the average of the selected features for \( M \) times and can be evaluated as Equation (21):

   \[
   \text{Mean} = \frac{1}{M} \sum_{i=1}^{M} \text{size}(x) \quad (21)
   \]

   where \( \text{size}(x) \) denotes to the selected feature size in the testing data set.

### Table 5: The classification accuracy for the proposed model and other approaches

| Dataset | Original dataset | PSO | GWO | BOA | Proposed model |
|---------|------------------|-----|-----|-----|----------------|
| DS_1    | 88.57            | 90.9| 90.24| 90.88| 97.1           |
| DS_2    | 96               | 95.7| 95.6 | 95   | 96.9           |
| DS_3    | 61.6             | 66.4| 63.3 | 67.2 | 70.6           |
| DS_4    | 87.1             | 85.4| 86.2 | 86.4 | 88.9           |
| DS_5    | 78.9             | 91.28| 90.26| 90.6 | 91.9           |
| DS_6    | 92.5             | 96.3| 96.4 | 96.5 | 97.3           |
| DS_7    | 89               | 92.5| 89.7 | 93.4 | 96.4           |
| DS_8    | 90.48            | 91.2| 90.4 | 91   | 95.2           |
| DS_9    | 85.1             | 87.6| 84.7 | 88.2 | 91.4           |
| DS_10   | 81.4             | 80.4| 82.1 | 82   | 82.9           |
| DS_11   | 95.4             | 93.5| 95.4 | 94   | 96.7           |
| DS_12   | 92.5             | 91.2| 90.4 | 91.4 | 93.9           |
| DS_13   | 68.8             | 72.1| 70.6 | 78.2 | 82.5           |
| DS_14   | 95               | 93.4| 90.2 | 92.4 | 95.8           |
| DS_15   | 71.5             | 83.4| 82.4 | 79.6 | 92.8           |
| DS_16   | 79               | 75.6| 81.2 | 79.2 | 85.3           |
| DS_17   | 84               | 89  | 89.4 | 91.3 | 95             |
| DS_18   | 72               | 80.2| 88.4 | 90.4 | 94.5           |
| DS_19   | 92.3             | 91.3| 92.8 | 90.4 | 98.2           |
| DS_20   | 95.2             | 96.5| 90.5 | 94   | 98.5           |
| DS_21   | 77               | 77.2| 76.8 | 77.1 | 79.3           |
| DS_22   | 92.5             | 93.4| 94.2 | 94.7 | 96.8           |
| DS_23   | 82.4             | 83.4| 84.3 | 82.7 | 82             |
| DS_24   | 86.2             | 90.7| 94.2 | 86.4 | 94.7           |
| DS_25   | 89.5             | 93.4| 95.2 | 94.2 | 98.2           |
| Average | 84.958           | 87.2792| 87.396| 87.8872| 91.076       |

5.1.2 Evaluation criteria

To evaluate the proposed BOAPSO, three measures are utilized as follows (Arora & Anand, 2019):

1. **Classification accuracy**: Is an indicator that explains how the classification is accurate given the set of features chosen; that is, the correct classification number of features and the precision of classification in this study is determined as Equation (20):

   \[
   \text{Average performance} = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{N} \sum_{i=1}^{N} \text{match}(C_i, L_i) \quad (20)
   \]

   where \( M \) is the number of times the optimization algorithm is performed, \( N \) indicates the number of test set points, \( C_i \) indicates the output class label of a unique Data point \( i \), \( L_i \) is a reference class label for \( i \) and corresponds to the comparison function, which gives output 1 if two labels are the same and output 0 otherwise.

2. **Average selection size**: Represents the average of the selected features for \( M \) times and can be evaluated as Equation (21):

   \[
   \text{Mean} = \frac{1}{M} \sum_{i=1}^{M} \text{size}(x) \quad (21)
   \]

   where \( \text{size}(x) \) denotes to the selected feature size in the testing data set.
3. **Average computational time**: Is the overall runtime of an individual optimization algorithm in seconds over different runs, and it can be calculated using Equation (22):

$$Mean = \frac{1}{M} \sum_{i=1}^{M} RunTime_{o,i}$$  \hspace{1cm} (22)

where $M$ is the number of runs for an optimization algorithm $o$, and $RunTime_{o,i}$ for the actual computational time for the optimization algorithm $o$ at run number $i$.

### 5.1.3 Parameters settings

Compared to standard PSO, standard BOA, and standard GWO, the efficiency of the proposed algorithm is compared with other common modern functional selection algorithms.

To ensure the contrast of the algorithms, the algorithms have been taken out of the literature. The KNN classifier is a popular wrapper for selecting features and is known as a kind of learning algorithm that is generally regulated by the simplicity and rapid implementation of the classification. Each algorithm with a random seed comprises 20 different runs. For the following studies, the maximum number of iterations is 20 for the $k$-fold cross validation norm (10-fold cross-validation). The data set was broken down into 10 folds in a ratio of 9:1 between the train and test data. For classification KNN with $K = 5$, training data is used while test data are kept separate. The number of search agents for the algorithm is 5. A solution is found in every search agent. Selected the best literature values for $\alpha$ and $\beta$ were performed on some of the data sets in several

| Dataset | Original dataset | PSO | GWO | BOA | Proposed model |
|---------|------------------|-----|-----|-----|----------------|
| DS_1   | 299              | 156.8 | 156.4 | 120.7 | 13.8          |
| DS_2   | 9                | 3.1  | 2.99 | 2.88 | 2.3           |
| DS_3   | 19               | 8.9  | 11.5 | 9.3  | 4.1           |
| DS_4   | 23               | 9.8  | 13.3 | 8.4  | 2.1           |
| DS_5   | 22               | 7.5  | 8.2  | 7.1  | 2.9           |
| DS_6   | 30               | 13.7 | 15.7 | 14.7 | 4.1           |
| DS_7   | 16               | 8.9  | 9.3  | 8.4  | 5.7           |
| DS_8   | 20               | 8.5  | 7.4  | 6.2  | 2.4           |
| DS_9   | 34               | 9.4  | 11.2 | 8.4  | 2.1           |
| DS_10  | 21               | 11.4 | 14.5 | 7.5  | 3.9           |
| DS_11  | 10               | 3.2  | 5.1  | 4.7  | 2.1           |
| DS_12  | 21               | 9.7  | 7.8  | 6.8  | 2.1           |
| DS_13  | 90               | 35.4 | 44.6 | 18.7 | 7.8           |
| DS_14  | 19               | 9.7  | 16.8 | 11.3 | 3.9           |
| DS_15  | 61               | 28.4 | 32.6 | 23.5 | 9.1           |
| DS_16  | 22               | 11.6 | 15.4 | 10.7 | 1.9           |
| DS_17  | 9                | 5.1  | 4.6  | 4.2  | 2.9           |
| DS_18  | 18               | 10.3 | 11.2 | 10.8 | 4.9           |
| DS_19  | 13               | 7.4  | 6.5  | 5.6  | 3.1           |
| DS_20  | 40               | 15.7 | 13.5 | 18.9 | 12            |
| DS_21  | 9                | 7.3  | 6.5  | 5.5  | 4.2           |
| DS_22  | 16               | 9.5  | 8.4  | 6.7  | 4.7           |
| DS_23  | 18               | 13.5 | 8.7  | 9.8  | 5.7           |
| DS_24  | 13               | 10.2 | 7.8  | 9.4  | 5.8           |
| DS_25  | 265              | 148  | 138.5 | 111.2 | 30.7          |
| Average| 44.68            | 22.52 | 23.1396 | 18.0552 | 5.764        |
observational experiments. $\alpha$ and $\beta$ are therefore values of 0.9 and 0.1, respectively. Table 3 describes the parameters of the proposed algorithm in collaboration with the GWO, PSO, and BOA.

### 5.1.4 The proposed BOAPSO results

Table 4 describes the results of the proposed approach as regards to classification accuracy, computational time, and the feature resulting after the irrelevant features have been eliminated. It is obvious that the proposed BOAPSO algorithm was far more efficient to classify accurately than the original data set. Figures 8 and 9 show a comparison between the classification accuracy of the proposed BOAPSO and the selected features as opposed to the accuracy and all the features from the original datasets.

### 5.1.5 Comparative algorithms

This section’s analysis illustrates that BOAPSO has superior performance in terms of classification accuracy, average selection size, and computational time compared to other approaches. The suggested model’s performance is compared to various state-of-the-art methods that are widely used in the literature to resolve the problem of feature selection. Regarding the accuracy of classification, Table 6 describes the initial dataset results, PSO, GWO, BOA, and the proposed BOAPSO. As shown in Table 5, all other approaches in all datasets are outperformed by the proposed model, clearly demonstrating the proposed approach’s strength. The native BOA is second in performance than GWO. Table 6 reports the average.

**Table 7** The computational time for the proposed model and other approaches

| Dataset | PSO | GWO | BOA | Proposed model |
|---------|-----|-----|-----|---------------|
| DS_1    | 55.9| 80.8| 45.6| 16.5          |
| DS_2    | 9.3 | 10  | 8.99| 7.3           |
| DS_3    | 9.9 | 8.4 | 9.4 | 5.6           |
| DS_4    | 7.5 | 6.9 | 6.8 | 6.1           |
| DS_5    | 7.9 | 8.5 | 7.8 | 5.11          |
| DS_6    | 8.7 | 8.4 | 6.9 | 5.9           |
| DS_7    | 6.5 | 7   | 6.4 | 4.3           |
| DS_8    | 9.4 | 8.9 | 7.6 | 6.5           |
| DS_9    | 9.4 | 8.8 | 7.4 | 6.2           |
| DS_10   | 11.7| 11.5| 10.5| 8.3           |
| DS_11   | 19.4| 17.5| 16.8| 12.2          |
| DS_12   | 9.4 | 7.4 | 8.5 | 4.5           |
| DS_13   | 9.5 | 8.7 | 7.5 | 6.2           |
| DS_14   | 8.7 | 9.5 | 7.8 | 5.1           |
| DS_15   | 8.7 | 9.6 | 8.4 | 5.8           |
| DS_16   | 7.8 | 9.6 | 8.4 | 6.2           |
| DS_17   | 17.5| 15.7| 11.4| 7.9           |
| DS_18   | 9.9 | 10.5| 9.8 | 6.2           |
| DS_19   | 9.2 | 8.7 | 8.7 | 6.9           |
| DS_20   | 13.2| 14.5| 13.4| 12.3          |
| DS_21   | 7.8 | 8.5 | 6.7 | 4.5           |
| DS_22   | 6.2 | 5.4 | 5.8 | 3.6           |
| DS_23   | 5.4 | 4.8 | 5.9 | 3.4           |
| DS_24   | 8.4 | 7.5 | 6.7 | 5.8           |
| DS_25   | 9.8 | 8.9 | 7.4 | 6.1           |
| Average | 11.5| 12.24| 10.0236| 6.7404        |
of the attributes selected using BOAPSO and other techniques. The BOAPSO shows significantly better performance on all datasets than other methods. This performance is underpinned by the increased capability to explore and exploit the proposed BOAPSO, which searches the feature space’s high-performance regions intensively.

FIGURE 10  The classification accuracy for the proposed model and other approaches

FIGURE 11  The selected features for the proposed model and other approaches
In Table 7, the proposed BOAPSO has the best computing performance on all datasets, while the general BOA was second in most data sets with higher performance and third is the PSO. Compared to state-of-the-art techniques, the proposed BOAPSO has shown competitive calculation speed. Consequently, the BOAPSO has been well performed relative to state-of-the-art methods in general.

All results that include the classification accuracy, the selected features, and the computational time are visualized in Figures 10 and 11, respectively. The convergence speed is the other factor in discussing, testing, and evaluating this recommended BOAPSO algorithm. The convergence curve based on the best fitness function and mean convergence curves for the proposed BOAPSO has been generated for three data set with high dimensionality to illustrate the effectiveness of the recommended BOAPSO, as seen in Figure 12. The proposed BOAPSO algorithm shows highly qualified performance from Figure 13 by inspecting the minimum fitness functions' convergence curves.

Compared with state-of-the-art approaches in terms of classification accuracy, the average number of attributes selected, and computational time, the proposed BOAPSO shows superior performance. BOAPSO is compared to better validate the performance of the proposed approach, with some newly developed techniques called fractional-order cuckoo search using heavy-tailed distributions (FO-CS) (Yousri et al., 2020) and the native Binary butterfly optimization approaches (Arora & Anand, 2019). Table 8b presents and visualizes the classification accuracy provided by the proposed model's selected features and comparative methods (b). It is easy to remember that in most of the standard data sets used in this research, the proposed (BOAPSO) over-performed all the other approaches. This finding demonstrates the ability of the BOAPSO to explore the search space and locate the ideal feature sub-set with the highest classification accuracy. The superior performance of the proposed BOAPSO can be found in Table 8a in terms of selecting the ideal function subset. The proposed approach outperformed all data sets by other algorithms, as seen in Figure 14a.

### 5.2 Experimental series 2: Feature selection using COVID-19 data-sets

The World Health Organization (WHO) declared in 2020 that the severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), known as COVID-19, had begun to strike China and spread exponentially worldwide. Also, COVID-19 has caused the deaths of over 600,000 individuals across the world since August 2020. Artificial intelligence has recently become the breakthrough in current technologies and can be used to the fight against COVID-19 for diagnosis, detection, and prevention (Too & Mirjalili, 2020). Feature selection is an essential task for healthcare. In COVID-19, the feature selection is necessary to determine the main attributes and features that provide an efficient decision for manipulating the patients. In this section, the proposed BOAPSO is employed for COVID-19 patient health prediction. The dataset of COVID-19 patients was collected and used from the GitHub data store (Novel Corona Virus 2019 Dataset, 2020). This dataset comprises 15 features, and 864 cases, the description of the dataset is shown in Table 9, and a sample of the transformed dataset is presented in Table 10.
This study intends to predict the death and recovery conditions depending on the given factors. All the features are converted into numeric form. Figures 15 and 16 demonstrate the accuracy and feature size of the proposed BOAPSO on the COVID-19 dataset, respectively. It is shown that BOAPSO achieved the highest classification accuracy of 95.4%. On the other hand, the result shows that roughly four features were enough for BOAPSO in patient health prediction. The most selected features were location, gender, age, and diff_sym_hos (the date patient started noticing the symptoms) based on the results obtained.

As a comparison to the state of, the proposed BOAPSO for COVID-19 feature selection and prediction model compared to the work in Too and Mirjalili (2020). The proposed model achieved 95.4% classification accuracy using four features, while in the related work achieved 92.1% using three features on the same dataset.

5.3 | Results and discussion

In this study, a binary version of BOAPSO is proposed and used to solve a feature-selection problem in the wrapper mode for a hybrid butterfly optimization algorithm (BOA) and the particle swarm optimization algorithm (PSO). The proposed model was applied to 25 datasets and compared to the native PSO, GWO, and BOA. The comparison included three metrics: classification accuracy, the selected feature set, and computational time. All results approved the superiority of the proposed model. The proposed model achieved a higher level of accuracy compared to other algorithms in most data sets based on classification accuracy. The KNN classifiers achieved an average precision of 91.7% for the proposed model instead of the comparative approaches with the native BOA of 87.8% in second place. In the Lymphographic dataset (DS_23), the native BOA achieved the best accuracy while the proposed model was the best in all other datasets.
The average of the proposed model’s total selected feature is 5.7, wherein other algorithms 18, 23, 23 and 45 for the BOA, GWO, PSO, and the original dataset, respectively. The percentage of the selected features for the proposed BOAPSO reached 12.7% from all the original features. The third metric is the computational time that showed the superiority of the proposed model in all datasets. The proposed model's selected features and classification accuracy for the proposed model, s-bBOA and FO-CS are as follows:

| Dataset | s-bBOA | FO-CS | Proposed model |
|---------|--------|-------|----------------|
| DS_2    | 5.6    | 3.4   | 2.3            |
| DS_7    | 5.2    | 6.1   | 5.7            |
| DS_9    | 16.2   | 15.5  | 2              |
| DS_15   | 32.8   | 36.8  | 9.1            |
| DS_16   | 10.8   | 5.3   | 1.9            |
| DS_19   | 6.2    | 5.7   | 3.1            |
| DS_20   | 25     | 26.3  | 12             |
| DS_21   | 5.6    | 6     | 4.2            |
| DS_22   | 5.2    | 6.6   | 4.7            |
| DS_23   | 8.4    | 11.6  | 5.7            |
| DS_24   | 7.6    | 6.8   | 5.8            |

The selected features and classification accuracy for the proposed model, s-bBOA and FO-CS are shown in the following figure:

![Selected Features](image1)

![Classification Accuracy](image2)

**FIGURE 14** The selected features and classification accuracy for the proposed model, s-bBOA and FO-CS. (a) Selected features. (b) Classification accuracy

The average of the proposed model’s total selected feature is 5.7, wherein other algorithms 18, 23, 23 and 45 for the BOA, GWO, PSO, and the original dataset, respectively. The percentage of the selected features for the proposed BOAPSO reached 12.7% from all the original features. The third metric is the computational time that showed the superiority of the proposed model in all datasets. The proposed model's selected features and classification accuracy for the proposed model, s-bBOA and FO-CS are as follows:
The model's average computational time equal 6.7 seconds while 11.5, 12.24, 10.02 for the PSO, GWO, and BOA, respectively. Compared to the most recent related work, the proposed model compared to s-bBOA and FO-CS using 11 standard datasets. The results showed the superiority of the proposed model in most datasets in classification accuracy, which greatly reduced the number of features. Besides the 25 dataset used for evaluating the proposed model, a COVID-19 dataset is used for patient prediction and approved of the proposed model's superiority.

### 5.4 Open research directions

There are various open issues reported in the research papers following an analysis of the solutions in the literature that most mimic the kind of innovations carried out. Some of them are as mentioned following:

1. Evolutionary algorithms (EAs) are usually stochastic search techniques based on population that share one algorithmic step, called population initialization. The role of this stage is to have an initial idea of solutions. These initially assumed solutions would then be iteratively modified during the optimization process before the stopping criteria is met. Generally, strong initial assumptions will make it easier for EAs to find the optimum. On the opposite, it can preclude EAs from finding the optimum starting from using poor guesses. This concern gets more critical when it comes to solving large-scale optimization problems using a finite size population. As population size is often small, the opportunity for a population to meet promising areas of the search space reduces as the size of the search space increases (Kazimipour et al., 2014).

2. The trade-off of exploration-exploitation is a well-known dilemma that arises in situations where a learning system must regularly make a decision of unknown payoffs. Exploration makes it possible, in one hand, to identify specific places in the search space and, on the other hand, manipulation makes it possible to maintain better options by searching the local search space. Among the metaheuristic search strategies listed above, some use the exploration approach, while others use the exploitation process for better returns. Consequently, the output of the search algorithm can be advanced by applying hybrid methods. Hybridization incorporates the positive features of at least two processes, thereby improving the yield of each procedure.

3. The values of the locations of the search agent created by the algorithm are continuous. Since it violates the common binary feature selection construction, it cannot be extended specifically to our problem. Based on the issue of the selection of features (0 or 1), the most suitable features are picked to improve the accuracy and efficiency of the classification algorithm. The calculated/resulting search space is converted into

| Column     | Description                          | Values                                                                 |
|------------|--------------------------------------|------------------------------------------------------------------------|
| `id`       | Patient id                           | Discrete numbers                                                      |
| `age`      | Patient's age                        | Different ages                                                         |
| `location` | The location where the patient belongs to | Multiple cities located throughout the world                             |
| `country`  | Patient's native country             | Multiple countries                                                     |
| `gender`   | Patient's gender                     | Male, Female                                                           |
| `Symptom1` | Fever                                | Multiple symptoms noticed by the patients                              |
| `Symptom2` | Cough                                |                                                                        |
| `Symptom3` | Cold                                 |                                                                        |
| `Symptom4` | Fatigue                              |                                                                        |
| `Symptom5` | Body pain                            |                                                                        |
| `Symptom6` | Malaise                              |                                                                        |
| `Sym_on`   | The date patient started noticing the symptoms | NA                                                                    |
| `Hosp_vis` | Date when the patient visited the hospital | NA                                                                    |
| `Diff_symp_hos` | Date when the patient visited the hospital - the date patient started noticing the symptoms | NA                                                                    |
| `Vis_wuhan` | Whether the patient visited Wuhan, China | Yes(1), No(0)                                                        |
| `From_wuhan` | Whether the patient belonged to Wuhan, China | Yes(1), No(0)                                                        |
| `death`    | Whether the patient passed away due to COVID-19 | Yes(1), No(0)                                                        |
| `Recov`    | Whether the patient recovered         | Yes(1), No(0)                                                          |
| Id | location | country | gender | age  | vis_wuhan | from_wuhan | symptom1 | symptom2 | symptom3 | symptom4 | symptom5 | symptom6 | diff_sym_hos | result |
|----|----------|---------|--------|------|-----------|------------|----------|----------|----------|----------|----------|----------|-------------|--------|
| 0  | 104      | 8       | 1      | 66   | 1         | 0          | 14       | 31       | 19       | 12       | 3        | 1        | 8           | 1      |
| 1  | 101      | 8       | 0      | 56   | 0         | 1          | 14       | 31       | 19       | 12       | 3        | 1        | 0           | 0      |
| 2  | 137      | 8       | 1      | 46   | 0         | 1          | 14       | 31       | 19       | 12       | 3        | 1        | 13          | 0      |
| 3  | 116      | 8       | 0      | 60   | 1         | 0          | 14       | 31       | 19       | 12       | 3        | 1        | 0           | 0      |
| 4  | 116      | 8       | 1      | 58   | 0         | 0          | 14       | 31       | 19       | 12       | 3        | 1        | 0           | 0      |
| 5  | 23       | 8       | 0      | 44   | 0         | 1          | 14       | 31       | 19       | 12       | 3        | 1        | 0           | 0      |
| 6  | 105      | 8       | 1      | 34   | 0         | 1          | 14       | 31       | 19       | 12       | 3        | 1        | 0           | 0      |
| 7  | 13       | 8       | 1      | 37   | 1         | 0          | 14       | 31       | 19       | 12       | 3        | 1        | 6           | 0      |
| 8  | 13       | 8       | 1      | 39   | 1         | 0          | 14       | 31       | 19       | 12       | 3        | 1        | 5           | 0      |
a continuous binary by a translation function. There are a variety of transformation functions, such as the V-shaped transfer function and the function of S-shaped transfer.

4. Linear modifications to the optimization algorithm parameters that vary in a linear way cannot represent the real optimization search process of the algorithm. But it is easier to adjust the control parameter nonlinearly by the number of iterations. And the typical test function optimization findings demonstrate that the use of nonlinear strategy is easier than linear strategy optimization.

6 | CONCLUSION

This paper presented a hybrid metaheuristic algorithm based on the standard butterfly optimization algorithm (BOA) and the standard particle swarm optimization algorithm (PSO) for the feature selection process. Three enhancement strategies are recommended to global optimize the basic BOA. The enhancements included the cubic map model’s initialization, the nonlinear power exponent control parameter, and the PSO’s use to enhance search capability in the BOA. To analyze the proposed model’s effectiveness, it compared with other swarm algorithms such as PSO, GWO, BOA, and other recent works using 25 datasets and a COVID-19 dataset are used. The initial HPSOBOA population had a cubic map sequence used, and the results of the tests showed that the initial fitness value was higher than the BOA and other algorithms.

Furthermore, the experimental results approved that one-dimensional chaotic maps can boost the standard BOA in improving its performance. The results supported the proposed model’s superiority in improving the classification process through the classification accuracy, the features selected, and the computational time. Future work involves improving the efficiency of the proposed algorithm and improving BOA by adapting its control parameters to maximize performance. The model proposed can also address other problems in reality, such as proportional-integral-derivative (PID) control problems, problems in engineering, regional economic activity research analysis, and the implementation problems of the wireless sensor network (WSN). Moreover, the butterfly can be hybridized with other MA like the salp swarm optimization algorithm. Besides, it is suggested that more clinical features can be obtained for accurate patient health prediction for COVID-19-patient-health-analytics.

CONFLICT OF INTEREST

The authors show no conflict of interest to submit this paper to this journal.
DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

ORCID

Mohamed Elhoseny https://orcid.org/0000-0001-6347-8368

REFERENCES

Abdel-Basset, M., El-Shahat, D., El-henawy, I. de Albuquerque Victor Hugo, C., & Mirjalili, S. (2020). A new fusion of Grey wolf optimizer algorithm with a two-phase mutation for feature selection. Expert Systems with Applications, 139, 112824.

Abualigha, L. M., & Khader, A. T. (2017). Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. The Journal of Supercomputing, 73(11), 4773–4795.

Abualigha, L. M., Khader, A. T., & Hanandehe, E. S. (2018). A new feature selection method to improve the document clustering using particle swarm optimization algorithm. Journal of Computational Science, 25, 456–466.

Al-Khafaji, M. H., & Abdulla Al-Kabrangi, S. S. (2011). Particle swarm optimization for total operating cost minimization in electrical power system. Engineering and Technology Journal, 29(12), 2539–2550.

Arora, S., & Anand, P. (2019). Binary butterfly optimization approaches for feature selection. Expert Systems with Applications, 116, 147–160.

Arora, S., & Singh, S. (2017). Node localization in wireless sensor networks using butterfly optimization algorithm. Arabian Journal for Science and Engineering, 42(8), 3325–3335.

Arora, S., & Singh, S. (2019). Butterfly optimization algorithm: A novel approach for global optimization. Soft Computing, 23(3), 715–734.

Arora, S., Singh, S., & Yetilmezsoy, K. (2018). A modified butterfly optimization algorithm for mechanical design optimization problems. Journal of the Brazilian Society of Mechanical Sciences and Engineering, 40(1), 21.

Bello, R., Gomez, Y., Nowe, A., & Garcia, M. M. (2007). Two-step particle swarm optimization to solve the feature selection problem. In Progress In Electromagnetics Research Conference on Intelligent Systems Design and Applications (ISDA 2007) (pp. 1200–1205). IEEE.

Brownlee, J. (2011). Clever algorithms: Nature-inspired programming recipes. Jason Brownlee.

Dheeru, D., & Taniskidou, E. K. (2017). [UCI] machine learning repository.

Emery, E., Zawbaa, H. M., & Hassani, A. E. (2016). Binary Grey wolf optimization approaches for feature selection. Neurocomputing, 172, 371–381.

Faris, H., Abukhurma, R., Almanaseer, W., Saadeh, M., Mora, A. M., Castillo, P. A., & Aljarah, I. (2020). Improving financial bankruptcy prediction in a highly imbalanced class distribution using oversampling and ensemble learning: A case from the Spanish market. Progress in Artificial Intelligence, 9(1), 31–53.

Golbon-Haghighi, M.-H., Saeidi-Manesh, H., Zhang, G., & Zhang, Y. (2018). Pattern synthesis for the cylindrical polarimetric phased array radar (CPPAR). Progress In Electromagnetics Research, 66, 87–98.

Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., & Yang, H. (2019). A generalized mean distance-based k-nearest neighbor classifier. Expert Systems with Applications, 115, 356–372.

Ibrahim, R. A., Ewees, A. A., Oliva, D., Abd Elaziz, M., & Lu, S. (2019). Improved Salp swarm algorithm based on particle swarm optimization for feature selection. Journal of Ambient Intelligence and Humanized Computing, 10(8), 3155–3169.

Imran, M., Hashima, R., & Khalid, N. E. A. (2013). An overview of particle swarm optimization variants. Procedia Engineering, 53, 491–496.

Jensen, R., & Shen, Q. (2004). Semantics-preserving dimensionality reduction: Rough and fuzzy-rough-based approaches. IEEE Transactions on Knowledge and Data Engineering, 16(12), 1457–1471.

Jovič, A., Brikč, K., & Bogunović, N. (2015). A review of feature selection methods with applications. In 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 1200–1205). IEEE.

Kabir, M. M., Shahjahan, M., & Murase, K. (2011). A new local search based hybrid genetic algorithm for feature selection. Neurocomputing, 74(17), 2914–2928.

Kazimipour, B., Li, X., & Qin, A. K. (2014). A review of population initialization techniques for evolutionary algorithms. In 2014 IEEE Congress on Evolutionary Computation (CEC) (pp. 2585–2592). IEEE.

Lal, T. N., Chapelle, O., Weston, J., & Elisseeff, A. (2006). Embedded methods. In Feature extraction (pp. 137–165). Springer.

Li, G., Shuang, F., Pan, Z., & Le, C. (2019). An improved butterfly optimization algorithm for engineering design problems using the cross-entropy method. Symmetry, 11(8), 1049.

Lu, H., Wang, X., Fei, Z., & Qiu, M. (2014). The effects of using chaotic map on improving the performance of multiobjective evolutionary algorithms. Mathematical Problems in Engineering, 2014, 1–16.

Mathiyalagan, P., Dhepthie, U. R., & Sivanandam, S. N. (2010). Grid scheduling using enhanced PSO algorithm. International Journal of Computational Science and Engineering, 2(2), 140–145.

Moradi, P., & Gholampour, M. (2016). A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. Applied Soft Computing, 43, 117–130.

Mosleh, F., & Haeri, A. (2020). A novel hybrid wrapper–filter approach based on genetic algorithm, particle swarm optimization for feature subset selection. Journal of Ambient Intelligence and Humanized Computing, 11(3), 1105–1127.

Novel Corona Virus 2019 Dataset. (2020). Retrieved from https://github.com/Atharva-Peshkar/Covid-19-Patient-Health-Analytics/blob/master/ final.csv

Pereira, G. (2011). Particle Swarm Optimization. INESC-ID Inst. Super. Techno Porto Salvo Port.

Qasim, O. S., & Algamal, Z. Y. (2018). Feature selection using particle swarm optimization-based logistic regression model. Chemometrics and Intelligent Laboratory Systems, 182, 41–46.

Rodrigues, D., de Albuquerque, V. H. C., & Papa, J. P. (2020). A multi-objective artificial butterfly optimization approach for feature selection. Applied Soft Computing, 94, 106442.

Rogers, T. D., & Whitley, D. C. (1983). Chaos in the cubic mapping. Mathematical Modelling, 4(1), 9–25.

Saccheri, I., Kuussaari, M., Kankare, M., Vikman, P., Fortelius, W., & Hanski, I. (1998). Inbreeding and extinction in a butterfly metapopulation. Nature, 392 (6675), 491–494.

Sadeghian, Z., Akbari, E., & Nematzadeh, H. (2021). A hybrid feature selection method based on information theory and binary butterfly optimization algorithm. Engineering Applications of Artificial Intelligence, 97, 104079.
AUTHOR BIOGRAPHIES

Dr. Ibrahim M. EL-Hasnony received his B.Sc., M.Sc., Ph.D degrees in information systems from Mansoura University, Egypt in 2010, 2016 and 2021, respectively. His research interests include cloud computing, big data, data analysis, smart city, the Internet of Things, neural networks, artificial intelligence, web service composition, blockchain, and evolutionary algorithms. He has published several papers in international and local conferences, journals and proceedings in different fields of information technology.

Dr. Mohamed Elhoseny is an Associate Professor of Computer Information Technology and the Manager of the Research Support Department at the American University in the Emirates, Dubai, UAE. Dr. Elhoseny is an ACM Distinguished Speaker and IEEE Senior Member. Dr. Elhoseny is the founder and the Editor-in-Chief of ISSTA Journal published by IGI Global. Also, he is an Associate Editor at several Q1 journals such as IEEE Journal of Biomedical and Health Informatics, IEEE Access, Scientific Reports, IEEE Future Directions, Remote Sensing, International Journal of E-services and Mobile Applications and Human-centric Computing and Information Sciences. Dr. Elhoseny is the Editor-in-Chief of the Studies in Distributed Intelligence Springer Book Series, the Editor-in-Chief of The Sensors Communication for Urban Intelligence CRC Press-Taylor& Francis Book Series, and the Editor-in-Chief of The Distributed Sensing and Intelligent Systems CRC Press-Taylor& Francis Book Series. He was granted several awards by diverse funding bodies such as the Egypt State Encouragement Award in 2018, the Young Researcher Award in Artificial Intelligence from the Federation of Arab Scientific Research Councils in 2019, Obada International Prize for young distinguished scientists 2020, Mansoura University Young Researcher Award 2019, the SRGE best young researcher award in 2017, and the best Ph.D. thesis in Mansoura University in 2015.

Dr. Zahraa Tarek received B.Sc degree in Computer Science from the University of Mansoura, Egypt, in 2009, and M.Sc in Computer Science from Mansoura University in 2015. In 2015, she joined, as assistant teacher, the Department of Computer Science, Mansoura University, and in 2017 she registered as a PhD research student in computer science department at faculty of computer and information system in the same university. She obtained the Ph.D. degree in Computer Science from the Faculty of Computers and Information in 2019. She has published 4 publications till now. Her current research interests include internet of things, fog computing, Blockchain, Artificial Intelligence, Machine Learning, Software Engineering.