Expert Systems With Applications 145 (2020) 113122

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection

Mohammad Tubishat[a], Norisma Idris[a,*], Liyana Shuib[b], Mohammad A.M. Abushariah[c], Seyedali Mirjalili[d]

[a] Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur 50603, Malaysia
[b] Department of Information Systems, Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur 50603, Malaysia
[c] Computer Information Systems Department, King Abdullah II School of Information Technology, The University of Jordan, Amman, Jordan
[d] Centre for Artificial Intelligence Research and Optimisation, Torrens University Australia, 90 Bowen Terrace, Fortitude Valley, QLD 4006, Australia

ARTICLE INFO

Article history:
Received 8 February 2019
Revised 28 August 2019
Accepted 3 December 2019
Available online 5 December 2019

Keywords:
Salp Swarm Algorithm
Classification
Feature selection
Optimization
Machine Learning
Algorithm
Opposition Based Learning

ABSTRACT

Many fields such as data science, data mining suffered from the rapid growth of data volume and high data dimensionality. The main problems which are faced by these fields include the high computational cost, memory cost, and low accuracy performance. These problems will occur because these fields are mainly used machine learning classifiers. However, machine learning accuracy is affected by the noisy and irrelevant features. In addition, the computational and memory cost of the machine learning is mainly affected by the size of the used datasets. Thus, to solve these problems, feature selection can be used to select optimal subset of features and reduce the data dimensionality. Feature selection represents an important preprocessing step in many intelligent and expert systems such as intrusion detection, disease prediction, and sentiment analysis. An improved version of Salp Swarm Algorithm (ISSA) is proposed in this study to solve feature selection problems and select the optimal subset of features in wrapper-mode. Two main improvements were included into the original SSA algorithm to alleviate its drawbacks and adapt it for feature selection problems. The first improvement includes the use of Opposition Based Learning (OBL) at initialization phase of SSA to improve its population diversity in the search space. The second improvement includes the development and use of new Local Search Algorithm with SSA to improve its exploitation. To confirm and validate the performance of the proposed improved SSA (ISSA), ISSA was applied on 18 datasets from UCI repository. In addition, ISSA was compared with four well-known optimization algorithms such as Genetic Algorithm, Particle Swarm Optimization, Grasshopper Optimization Algorithm, and Ant Lion Optimizer. In these experiments four different assessment criteria were used. The rdemonstrate that ISSA outperforms all baseline algorithms in terms of fitness values, accuracy, convergence curves, and feature reduction in most of the used datasets. The wrapper feature selection mode can be used in different application areas of expert and intelligent systems and this is confirmed from the obtained results over different types of datasets.

1. Introduction

Recently, data become an important source for many fields such as data mining and data science. However, the rapid growth of data volume will result in many problems such as the emergence of noisy, high dimensionality or irrelevant data. These problems decrease the machine learning classification accuracy and increase computational costs. In addition, all traditional machine learning classifiers are unable to interact with features contained in the used datasets. Therefore, these mentioned problems can affect the performance of data mining, and data science fields because it is mainly used machine learning classifiers. Thus, feature selection is mandatory required to select the most informative features and removes the irrelevant features

Feature selection has been used in many expert and intelligent systems such as in intrusion detection (Acharya & Singh, 2018; Aljawarneh, Aldwairi & Yassein, 2018; Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee & Karimipour, 2019; Selvakumar et al., 2019; Vijayanand, Devaraj & Kannapiran, 2018), image processing applications (Adeli & Broumandnia, 2018; Chen, Tondi et al., 2019; Cheng et al., 2016; Damodaran, Courty & Lefèvre, 2017; Jothi, 2016;

* Corresponding author.
E-mail addresses: mtubishat@siswa.um.edu.my (M. Tubishat), norisma@um.edu.my (N. Idris), liyanashuib@um.edu.my (L. Shuib), m.abushariah@ju.edu.jo (M.A.M. Abushariah), ali.mirjalili@gmail.com (S. Mirjalili).

https://doi.org/10.1016/j.eswa.2019.113122

Wang, Wu, Wang, Xiang & Huang, 2019; Yao, Liu, Jiang, Han & Han, 2017), sentiment analysis (Ahmad, Bakar & Yaakub, 2015, 2019; Alarifi, Tolba, Al-Makhadmeh & Said, 2018; Kumar, Jaiswal, Garg, Verma & Kumar, 2019; Pratiwi, 2018; Tubishat, Abushariah, Idris & Aljarah, 2019), cancer detection (González, Castillo, Galvez, Rojas & Herrera, 2019; Jain, Jain & Jain, 2018; Rani & Devaraj, 2019; Sawhney, Mathur & Shankar, 2018; Sayed, Nassef, Badr & Farag, 2019; Zhang, Zou, Zhou & He, 2018), Software fault prediction (Turabieh, Mafarja & Li, 2019), disease detection and classification (Cheruku, Edla, Kuppili & Dharavath, 2018; Gupta et al., 2018; Jayaraman & Sultana, 2019; Mirzaei et al., 2018; Sharif et al., 2018; Sharma, Sundaram, Sharma, Sharma & Gupta, 2019), design problems (Hasanloei, Sheikhpour, Sarram, Sheikhpour & Sharifi, 2018), power systems (Bagheri et al., 2018; Beyranvand, Kucuktezcan, Cataltepe & Genc, 2018; Chen, Han, Fan, Zheng & Mei, 2019; Ghadimi, Akbarimajd, Shayeghi & Abedinia, 2019; Ranjbar & Jamali, 2019; Yang, Liu, Tao & Hu, 2018), data mining (Dutta, Ghatak, Das, Gupta & Dasgupta, 2019), text classification (Guru, Ali & Suhil, 2019), and many more other application areas.

For example, in intrusion detection system (IDS) the main problem is the task of handling huge volume of data to accurately detecting network intrusions. This is huge data result because there is a huge number of network transactions. However, using all features in the used datasets will result in increasing the computational cost which is required for training the IDS model. In addition, it reduces the IDS prediction accuracy (Aljawarneh et al., 2018). The sentiment analysis at document and sentence levels are mainly worked using machine learning classifiers. The huge volume of available data on shopping websites such as Amazon affects the accuracy of sentiment analysis and increase computational and memory cost. Therefore, feature selection is required to improve sentiment analysis performance and reduce computational and memory cost (Tubishat, Abushariah, Idris & Aljarah, 2018). Image processing application mainly affected by the dimensionality of image features. Therefore, the availability of redundant and irrelevant features in the image representation can result in increasing the computational and memory cost of image applications. To solve these problems feature selection methods are typicaly used in the literature (Adeli & Broumandnia, 2018). In cancer prediction or classification, the existence of irrelevant or redundant genes expressions will result in lower accuracy and more computational cost. Therefore, feature selection will be used to select the most important gene features while it is maximizing the classification accuracy (Jain et al., 2018).

Feature selection methods are mainly classified into two main types: filter-based methods and wrapper-based methods. In filter-based methods (e.g., Information Gain, Gini Index, Chi-Square, and Relief), the features are ranked and selected based on statistical methods. In addition, filter-based method has no direct interaction with the machine learning classifier and it is conducted before applying the classifier (Jensen & Shen, 2008). On other hand, wrapper-based method work using optimization algorithm with machine learning classifier for feature selection. In the wrapper mode, there is a direct interaction with the used classifier. Filter-based feature selection methods are computationally cheaper and faster than wrapper-based methods. However, wrapper-based methods are more commonly used in feature selection problem because it is more beneficial in terms of improving classifier accuracy and features reduction because it has a direct contact with the used classifier (Liu & Yu, 2005).

Wrapper-based methods used fitness function to evaluate the selected features based on the classification accuracy (Kohavi & John, 1997). To improve classification accuracy, several studies have been conducted using optimization algorithms such as works which were conducted in Aljarah et al. (2018), Arora and Anand (2019), Gu, Cheng and Jin (2018), Mafarja and Mirjalili (2018), Marie-Sainte and Alalyani (2018), Selvakumar and Muneeswaran (2019), Tubishat et al. (2018), Zhang, Mistry, Lim and Neoh (2018). The main reason of using optimization algorithms in feature selection is because it can return a solution that is closer to optimal solution or the optimal solution within acceptable time. On other hand, normal exhaustive search works by trying all possible features combination from the full feature set, which is time consuming search and considered as an NP hard problem (Guyon & Elisseeff, 2003). Therefore, an optimization algorithm is required to solve this issue in acceptable time because of its capability to arrive at solution that can be considered as optimal solution or near to optimal (Kohavi & John, 1997).

Recently, many optimization algorithms have been developed and applied to different types of applications. For example, Sayed, Hassanien and Azar (2017) improved Crow search algorithm (CSA) using chaotic theory for feature selection problems. Kuo, Huang, Zulvia and Liao (2018) applied artificial bee colony (ABC) optimization algorithm for feature selection. Emary, Zawbaa and Hassanien (2016) used grey wolf optimizer (GWO) for feature selection. Eesa, Orman and Brifcani (2015) used Cuttlefish algorithm (CFA) for feature selection in IDS. Furthermore, Bat Algorithm (BA) was used by Rodrigues et al. (2014) in feature selection. Mafarja and Mirjalili (2018) improved Whale Optimization Algorithm (WOA) for feature selection. (Gu et al., 2018) used competitive swarm optimizer (CSO) for feature selection. Aljarah et al. (2018) used Grasshopper optimization algorithm (GOA) for feature selection. Acharya and Singh (2018) used intelligent water drops (IWD) for selecting the most informative features for IDS. Hancer, Xue and Zhang (2018) used differential evolution (DE) algorithm for feature selection.

SSA algorithm is a recent swarm optimization algorithms. SSA proved its superiority and outperformance in comparison to other well-known optimization algorithms over several types of problems. In addition, SSA has number of characteristics such as simplicity and flexibility. As a stochastic optimization algorithm, however, SSA suffers from some problems such population diversity and local optima. Based on No-Free-Lunch (NFL) theorem (Wolpert & Macready, 1997), there is no single feature selection optimization algorithm that can always select the optimal subset of features. In addition, none of the available optimization algorithms can always be superior to all other optimization algorithms in terms of accuracy over all available datasets. This means that a specific feature selection optimization algorithm may outperform other algorithms over a subset of datasets but not all available types of datasets. These mentioned reasons and characteristics motivated our study to improve SSA. In addition, to use an improved SSA (ISSA) to select the optimal subset of features while getting better classification accuracy.

Salp Swarm Algorithm (SSA) developed by Mirjalili et al. (2017). SSA mimic the slaps swarming behaviors which include slaps foraging and navigation inside the oceans. From the conducted experiments in Mirjalili et al. (2017), it was confirmed SSA ability and outperformance in comparison with other well-known optimization algorithms. In addition, SSA was used in many applications and proved its superiority. For example, SSA was used in the following applications, for feature selection on biomedical dataset (Ibrahim, Mazher, Ucan & Bayat, 2017), seismic exited structural system (Baygi, Karsaz & Elahi, 2018), for parameter estimation of the soil water retention curve model (Zhang, Wang & Luo, 2018), for locating and finding the optimal size of renewable distributed generators (RDGs) (Tolba, Rezk, Diab & Al-Dhaifallah, 2018), for segmentation of fish image (Ibrahim, Ahmed, Hussein & Hassanien, 2018), for finding the optimal allocation of shunt capacitors and distributed generations (Asasi, Ahanch & Holari, 2018), for parameter tuning and optimization in power system (Ekinci & Hekimoglu, 2018), and to find the optimal parameters values for fuel cells (El-Fergany, 2018). These findings motivated this study to further apply SSA for feature selection problems. However, SSA like other optimization algorithms suffers from the problem of population diversity and trapping into locally optimal solutions. Therefore, two main improvements were included into the original SSA algorithm to solve its problems. The first improvement includes the use of OBL strategy at the initialization phase of SSA to improve its population diversity. The second improvement includes the development of new local search algorithm which will be used at the end of each SSA iteration. This LSA algorithm will be used to improve SSA exploitation and avoid it from being trapped at local optima.

Although the PSO algorithm has number of benefits such as its simplicity and fast convergence speed. However, this algorithm has number of limitations such population diversity and local optima (Qiu, 2019). In addition, PSO does not work effectively for large scale problems (Gu et al., 2018). GA algorithm has limitations such as the problem of crossover operator which is suddenly changed the solutions through the search process (Mafarja & Mirjalili, 2018). GOA has limitations such as slow motion, small grasshoppers steps, and premature convergence (Ewees, Elaziz & Houssein, 2018). ALO suffers from the problems of local optima and premature convergence (Emary & Zawbaa, 2019).

In comparison to other known optimization algorithms such as GA and PSO, SSA has a fewer number of parameters and is easy to implement (Hegazy, Makhlouf & El-Tawel, 2019). Furthermore, SSA confirmed its ability to solve large-and-small-scale problems (Mirjalili et al., 2017). SSA featured with its flexibility and strongly stochastic nature (Faris et al., 2018). Although, SSA has these interesting benefits but as other optimization algorithms, SSA suffers from some problems as mentioned earlier. Therefore, the proposed ISSA inherits the features of original SSA algorithm. In addition, ISSA superior to original SSA for its ability to balance between exploitation and exploration, and avoid local optima based on the proposed improvements.

OBL had been used in many studies in literature to improve different optimization algorithms. OBL proved its ability to boost these algorithms performance. For example, Rahnamayan, Tizhoosh and Salama (2008) used OBL to improve the convergence speed of differential evolution (DE). Shan, Liu and Sun (2016) used OBL to improve the population diversity and convergence speed of Bat Algorithm (BA). Sapre and Mini (2018) used OBL to improve the convergence speed of moth flame optimization (MFO). Zhou, Hao and Duval (2017) used OBL to improve the convergence speed and population diversity of memetic algorithm (MA). Sarkhel, Chowdhury, Das, Das and Nasipuri (2017) used OBL to improve the convergence speed of Harmony Search (HS). Verma, Aggarwal and Patodi (2016) used OBL to improve the convergence speed of Firefly Algorithm (FA). Bairathi and Gopalani (2018) used OBL to improve exploration of SSA and used it for numerical optimization. However, in their study they used OBL with every iteration in SSA. Therefore, this will increase the complexity of the search algorithm because in each iteration they determine the opposite solutions for all solutions and take elite solutions. In our work, we use OBL at SSA initialization phase only. Zhou, Fang, Wu, Sun and Cheng (2016) used OBL to improve the convergence speed of competitive particle swarm optimizer (CPSO). Based on the improved performance resulted from combining OBL with these mentioned optimization algorithms, this study proposed to use OBL with SSA to improve its population diversity.

As mentioned earlier, to overcome the problems associated with original SSA, an improved SSA (ISSA) is proposed based on LSA and OBL strategy. In addition, ISSA algorithm works in wrapper-based feature selection to improve classification accuracy. To achieve this main aim the following contributions are proposed.

In this work, there are number of contributions which are outlined as the following:

- ISSA: An improved version of the standard SSA is proposed to solve the problems of standard SSA.
- Two major improvements were included into SSA using OBL strategy and Local Search Algorithm (LSA)
- OBL: OBL is used with SSA to improve its population diversity in the search process.
- LSA: new LSA algorithm is proposed and developed in this study.
- LSA: LSA is combined with SSA to solve its local optima problem and to improve the current best solution.
- ISSA wrapper feature selection model is proposed in this study.
- We compared ISSA algorithm with number of algorithms including (SSA, GA, PSO, GOA, and ALO) using 18 benchmark datasets from UCI repository. From these comparisons results it is clearly noticed the outperformance of ISSA to these optimization algorithms in terms of fitness values, accuracy, convergence speed and feature reduction.

The rest of this study are organized as follows: Section 2 presents some preliminaries about SSA and other used ideas. Section 3 presents the details of ISSA algorithm based on the introduced improvements. Section 4 presents the details about the different experiments were used to verify the performance of proposed ISSA algorithm. Finally, we conclude the paper in Section 5.

2. Preliminaries

2.1. Salp Swarm Algorithm

SSA is a new optimization algorithm developed by Mirjalili et al. (2017). SSA mimics the swarming behavior of salps in nature. Salps belong to big family of animals known as Salpidae's family. Salps are featured with barrel bodies and transparent body like jelly fishes. In addition, salp move is like the movement technique used by jelly fish. The swarming behavior of the salps is the main interesting technique, where the salps use their swarming behavior in the deep oceans to form a chain which is known as salp chain. This salp chain was modeled by Mirjalili et al. (2017) as an optimization algorithm, where they divided the population into two groups: followers and leaders. In SSA algorithm, the leader represents the chain front, while other salps in the chain represent the followers. The followers in the chain follow each other and this chain guided by the leader (Mirjalili et al., 2017). Mathematically, the position of each salp in the population of SSA is defined in n dimensions search space, where n is the number of variables according the problem to be solved. Moreover, x is a matrix with two-dimensions to save the positions of all salps. In addition, food source F represents the swarm target (Mirjalili et al., 2017). Eq. (1) is utilized in SSA algorithm to update the leader-position (Mirjalili et al., 2017):

$$x_j^1 = \begin{cases} F_j + c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & c_3 \geq 0.5 \\ F_j - c_1\left(\left(ub_j - lb_j\right)c_2 + lb_j\right) & c_3 < 0.5 \end{cases} \tag{1}$$

In this equation, the leader position in slaps chain is represented by x_j^1 value, and F_j shows the food source position in j^{th} dimension. Furthermore, lb_j indicates the lower bound value of j^{th} dimension and ub_j represents the upper bound value of j^{th} dimension. Also, c_1, c_2, and c_3 are random values, where c_1 is the most important controlling parameter in SSA because it is responsible of making balance between SSA exploitation and exploration. The following Eq. (2) used to calculate the value of c_1 parameter

```
Initialize the salps positions x_i (i = 1, 2, ..., n)
while (t < max iterations)
        determine the fitness value of each salp
        F = best salp ((search-agent)
        Update the value of c_1 parameter using Equation (2)
        for every salp (x_i)
            if (i ==1)
                Update leader position using Equation (1)
            else
                Update follower position using Equation (3)
            end if
        end for
        reposition the salp which go out search space based on lower and upper bounds of
        problem variables
        t=t+1
end while
return F
```

Fig. 1. SSA algorithm (Mirjalili et al., 2017).

(Mirjalili et al., 2017).

$$c_1 = 2e^{-\left(\frac{4t}{L}\right)^2} \tag{2}$$

In Eq. (2) t represents the current-iteration and L represents maximum number of iterations of the algorithm. The c_2 and c_3 parameters are random values in the interval [0,1]. To update the positions of the followers' in the slaps chain, Eq. (3) will be used (Mirjalili et al., 2017):

$$x_j^i = \frac{1}{2}\left(x_j^i + x_j^{i-1}\right) \tag{3}$$

where $i \geq 2$ and x_j^i contains the position of the i^{th} follower in the j^{th} search-space dimension. The pseudocode of the original SSA algorithm is shown in Fig. 1. SSA starts by generating its populations positions (salps) randomly. Then, SSA will evaluate the fitness values of all salps. After that, SSA considers the best (fittest-salp) solution as the food Source (*F)* which represents the target of the followers in the chain. Moreover, in each iteration, SSA will update the value of c_1 parameter using Eq. (2), while leader position will be updated using Eq. (1). Also, followers' positions will be updated using Eq. (3). All of SSA steps except initialization step will be repeated until SSA reaches the maximum number of iterations.

2.2. Opposition based learning (OBL)

OBL represents an optimization technique which was used by many studies to improve the quality of their initiated population solutions by diversify these solutions. The OBL strategy works by search both directions in search space. These two directions include one the original solution while the other direction is represented by its opposite solution. Finally, the OBL strategy takes the fittest solutions from all solutions.

- **Opposite number:** x is defined as a real number over the interval $x \in [\,lb,\ ub]$. The opposite number of x is denoted by $\tilde{x}$ and to determine its value Eq. (4) will be used [27]:

$$\tilde{x} = lb + ub - x \tag{4}$$

Eq. (4) can be generalized to apply it in a search space with multidimensions. Therefore, to generalize it, every search-agent position and its opposite position will be represented by the following Eqs. (5) and (6):

$$x = [x_1, x_2, x_3, \ldots, x_D] \tag{5}$$

$$\tilde{x} = [\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \ldots, \tilde{x}_D] \tag{6}$$

The values of all elements in $\tilde{x}$ will be determined using Eq. (7):

$$\tilde{x}_j = lb_j + ub_j - x_j \text{ where } j = 1, 2, 3, \ldots, \mathrm{D} \tag{7}$$

- **Optimization Based on Opposite population**: In this strategy the fitness function is $f(.)$. Therefore, if the fitness value $f(\tilde{x})$ of the opposite solution is superior than $f(x)$ of its original solution x, then $x = \tilde{x}$; otherwise $x = x$.
- **The procedure for integrating OBL with SSA is summarized by the following**:
  1. Initialize the salps positions X as x_i *where (i = =1, 2, …, n).*
  2. Determine the opposite positions of salps population OX as $\tilde{x}_i$ *where (i = =1, 2, …, n).*
  3. Select the n fittest salps from {X ∪ OX} and this now represents the new initial population of SSA.

2.3. Local search algorithm (LSA)

The new proposed and developed LSA algorithm is shown in Fig. 2. LSA will be called at the end of each iteration in SSA, to enhance the current F best solution. At first, LSA starts by storing the value of best solution F which is received from SSA at the end of SSA iteration in *Temp* variable.

LSA iterates number of times trying to improve *Temp*. In each iteration of LSA, LSA selects three features randomly from *Temp*. LSA sets or resets the selected features based on its values. Moreover, LSA will determine the fitness value of the new solution, if it is better than the fitness value of F, then F will be set to *Temp*; othewise F remains unchanged.

3. Proposed Improved Salp Swarm Algorithm (ISSA)

This section presents the two improvements which are included into the standard SSA algorithm as shown in Fig. 3. The first improvement includes the use of OBL strategy at initialization phase of SSA to improve its population diversity. The second improvement includes the use of LSA algorithm with SSA to improve its exploitation and avoid it from stuck into local optima. As shown in Fig. 3, ISSA works at first by generating salps population using OBL. Then, it takes the n fittest salps form the initial salps and opposite salps positions. In addition, the best salp among these n fittest salps will be set as F. Furthermore, the main loop will be applied on these n salps to update their positions using either Eq. (1) or (3). Now, at the end of ISSA main loop, LSA will be applied on current F to check and find better solution than current best solution. At end, ISSA will return the best solution F.

```
Temp= F  (where F represent the current best solution at end of SSA current iteration)
Lt=1 (Lt is a variable used to store the current iteration of local search algorithm)
while (Lt < maximum number of local iterations)
        Randomly select three features from Temp
        if  selected-feature == 1    (1 mean the feature is selected and 0 means not selected)
                selected-feature= 0
        else
                selected-feature=1
        end if
        Calculate the fitness value of Temp
        if  f(Temp) < f(F)
            F=Temp
        end if
        Lt= Lt+1
end while
return F
```

Fig. 2. LSA algorithm.

```
Initialize the salps positions X as x_i (i = 1, 2, ..., n)
Calculate the salps opposite population OX as x̃_i  (i = 1, 2, ..., n) using Equation (7)
Select the n fittest salps from {X ∪ OX} which represent now the initial SSA population
determine the fitness value of each salp
F = best salp ((search-agent)
while (t < max iterations)
        Update the value of c_1 parameter using Equation (2)
        for every salp (x_i)
            if (i ==1)
                Update leader position using Equation (1)
            else
                Update follower position using Equation (3)
            end if
        end for
        reposition the salp which go out search space based on lower and upper bounds of  problem
        variables
        determine the fitness value of each salp
        F = best salp ((search-agent)
        Apply LSA on F to find if there is a better solution (if better solution found then update F ;
        otherwise F left unchanged)
        t=t+1
end while
return F
```

Fig. 3. Proposed Improved ISSA algorithm based on OBL and LSA algorithm.

ISSA for wrapper feature selection mode: This proposed ISSA algorithm works with KNN classifier using wrapper mode for feature selection problems. In each iteration, ISSA will be applied on the training dataset to find the features subset for training the K-Nearest Neighbor (KNN) classifier. Binary values are used for the representation of selected and unselected features in feature selection problem, such that the existence of "1″ value in the solution indicates that features with the given index is selected, while the existence of "0″ value in the solution indicates that features with the given index is unselected. The steps of the proposed ISSA algorithm are as follows:

1. **ISSA initialization:** In this step, ISSA randomly generates number of salps based on the population size. Moreover, each generated solution contains a subset of features which are randomly selected from the whole set of features.
2. **Apply OBL**: In this step, OBL used to find the opposite solution of each solution in step 1. Then, OBL selects the n fittest solutions form the set of initial population in step 1 and its opposite solutions which are determined in step 2. (ISSA calculates the fitness values based on KNN classification accuracy error). Furthermore, from the best solutions which were selected by OBL, ISSA assigns F value with the best solution among these solutions, which represents the solution with lowest classification accuracy error.
3. **Update salps positions:** In this step, ISSA updates the positions of each salp using either Eq. (1) or (3). If the current salp is the leader of salps chain, then Eq. (1) will be used to update its position; otherwise Eq. (3) will be used to update the followers' position.
4. **Fitness evaluation:** In this step ISSA will determine the fitness values of all salps. Then, ISSA will update the value of F if there is a better solution.
5. **Apply LSA algorithm:** LSA will be applied on the current best solution F to find better solution. Then, LSA will update the value F if it finds better solution.
6. **ISSA execution termination:** The proposed ISSA algorithm repeats steps 3,4, and 5 t times.
7. **Solution:** The best solution F will be returned by ISSA and this represents the best subset features which were selected by ISSA.
8. **Testing Phase:** in this phase, the features which are selected in the best solution will be used to evaluate the performance of ISSA on the testing dataset.

Table 1
Details of the used datasets.

#	Dataset	Number of features	Number of instances
1	exactly	13	1000
2	exactly2	13	1000
3	m-of-n	13	1000
4	sonar	60	208
5	ionosphere	34	351
6	credit	20	1000
7	liver	6	345
8	spambase	57	4601
9	heart	13	270
10	lymphography	18	148
11	spect	22	267
12	tic-tac-toe	9	958
13	vote	16	300
14	Australian	14	690
15	waveform	21	5000
16	Vertebral	6	310
17	dermatology	34	366
18	Satellite	36	6435

Table 2
The Parameters setting of optimization algorithms.

Algorithm	Parameter
PSO	Acceleration-constants values are C1 = 1.5 and C2 = 2
	Inertia-Weight values are W1 = 1 and W2 = 0.9
GOA	cMax value is 1
	cMin value is 0.00004
GA	Crossover-ratio value is 0.8
	Mutation-ratio value is 0.2

4. Experimental results and analysis

4.1. Datasets

All algorithms were implemented using MATLAB. Moreover, to evaluate and validate the performance of ISSA algorithm in comparison with other algorithms, 18 UCI benchmark datasets from the UCI datasets repository were used in all experiments (Dua & Taniskidou, 2017). These datasets were used by majority of researches in feature selection. The details of the used datasets are present on Table 1.

4.2. Parameter settings

In all conducted experiments, we used 10-fold cross_validation over all experiments. This validation works by using nine folds from these ten folds for training phase, while one fold will be used for testing. In addition, the experiment was carried and repeated for ten times. In result, the average values over the ten experiments were reported over all used performance metrices. Therefore, the results in the following tables represent the average values over 10 runs in terms of accuracy, fitness, and the selected features. We compared our proposed ISSA algorithm with the standard SSA and other state-of-the-art optimization algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Lion Optimizer (ALO), and Grasshopper Optimization Algorithm (GOA). The parameter settings for each algorithm are shown in Table 2. Moreover, the population size which used for all algorithms is 10, the maximum number of iterations for each optimization algorithm is 40, and the number of LSA iterations was set to 10. The main metric for evaluating the optimization algorithms performance is the classification accuracy, where the used classifier is KNN classifier.

4.3. Results and analysis

This section outlines the details and results of all experiments. The first experiment includes the comparison of the proposed ISSA with the original SSA. In the second experiment, ISSA algorithm was compared with other algorithms such as GA, PSO, GOA, and ALO. The comparison between ISSA and other algorithms based on using three metrices as the following, where in each metric the average value was reported over 10 runs:

- Classification accuracy
- Number of selected features
- Fitness value

4.3.1. Comparison of SSA and ISSA

In this experiment, the original SSA was compared with the proposed ISSA algorithm. As shown in Table 3, which presents the classification accuracy of both algorithms over all 18 datasets, ISSA algorithm outperforms original SSA algorithm over all 18 datasets. Thus, ISSA is performing much better than the original SSA algorithm. Moreover, based on the number of selected features, as shown in Table 3, ISSA outperformed SSA algorithm because ISSA was selected fewer number of features over all 18 datasets. Lastly, ISSA outperformed SSA in term of fitness values over all 18 datasets because the classification error of ISSA is less than SSA algorithm. From these reported results based on classification accuracy, number of selected features, and fitness value, its clearly demonstrated that ISSA is significantly improved the performance further than the original SSA algorithm.

4.3.2. Comparison of ISSA with other optimization algorithms including GA, PSO, GOA, and ALO

In previous experiments, ISSA was compared with the original SSA algorithm. ISSA demonstrated a superior performance over standard SSA algorithm. Moreover, these improvements in the performance results achieved because of ISSA ability to balance between exploration and exploitation, enhancement of population diversity, and ability to escape from local optima. To further confirm the superiority of ISSA algorithm to other optimization algorithms in literature, ISSA was compared to GA, PSO, GOA, and ALO algorithms. Table 4 shows the comparisons of classification accuracy between ISSA and other algorithms. From Table 4, it is observed that ISSA outperformed all other algorithms in term of classification accuracy over all 18 datasets.

Table 5 presents the average number of selected features over 10 runs by each algorithm. It is evident that ISSA outperformed other optimization algorithms over 14 datasets out of 18 datasets in features reduction as indicated by bold font, while also it outperforms original SSA over all 18 datasets. Furthermore, GA outperformed other algorithm over one dataset only, PSO outperformed other algorithms only over one dataset only, GOA outperformed other algorithms over one dataset only and ALO outperformed other algorithms over two datasets only. The superiority of ISSA algorithm in term of selecting fewer number of features is proved by using LSA and OBL strategy. Moreover, as ISSA can select fewer number of features, this means it can explore the most important search space areas and avoid searching among irrelevant space areas. Therefore, ISSA can reduce the search space size by selecting the most informative features, while it is also improving the classification accuracy.

Based on the obtained fitness values of ISSA in comparison with other optimization algorithms as presented in Table 6, it is confirmed the superiority of ISSA to other optimization algorithms over all 18 datasets. In addition, the used datasets include both large and small size datasets, which proves the capability of ISSA to perform in consistency over datasets regardless of dataset size.

Table 3
Comparison between ISSA and SSA based on average-accuracy, average-number-of-selected- features, and average-fitness in 10 runs.

Dataset	Accuracy		Number of selected features		Fitness	
	SSA	ISSA	SSA	ISSA	SSA	ISSA
exactly	0.934	**1**	7.2	**6.4**	0.070	**0.004**
exactly2	0.76	**0.784**	6.6	**6.2**	0.242	**0.218**
m-of-n	0.96	**1**	7.6	**6.5**	0.045	**0.005**
sonar	0.961	**0.990**	26.4	**16**	0.042	**0.011**
ionosphere	0.971	**0.994**	13.9	**10.7**	0.031	**0.007**
credit	0.746	**0.798**	9.4	**8.8**	0.256	**0.204**
liver	0.730	**0.739**	2.9	**2.8**	0.271	**0.262**
spambase	0.934	**0.950**	29.5	**27.4**	0.070	**0.053**
heart	0.859	**0.881**	5.6	**4.6**	0.143	**0.120**
lymphography	0.681	**0.825**	7.9	**7.4**	0.319	**0.176**
spect	0.816	**0.865**	10.5	**9.6**	0.186	**0.137**
tic-tac-toe	0.829	**0.837**	5.6	**5.5**	0.175	**0.167**
vote	0.983	**0.993**	6.2	**5.1**	0.019	**0.009**
Australian	0.840	**0.878**	5.9	**4.7**	0.161	**0.123**
waveform	0.802	**0.815**	12.6	**12.3**	0.202	**0.189**
Vertebral	0.887	**0.890**	3.6	**3**	0.118	**0.113**
dermatology	0.991	**1**	15	11.2	0.011	**0.002**
Satellite	0.918	**0.930**	19.7	**18.1**	0.086	**0.074**

Table 4
ISSA Comparison with other algorithms based on average classification accuracy in 10 runs.

Dataset	ISSA	GOA	PSO	ALO	GA
exactly	**1**	0.728	0.875	0.71	0.909
exactly2	**0.784**	0.743	0.78	0.718	0.756
m-of-n	**1**	0.846	0.952	0.831	0.979
sonar	**0.990**	0.903	0.966	0.903	0.975
ionosphere	**0.994**	0.928	0.974	0.937	0.965
credit	**0.798**	0.715	0.754	0.69	0.764
liver	**0.739**	0.704	0.733	0.669	0.715
spambase	**0.950**	0.906	0.938	0.904	0.934
heart	**0.881**	0.788	0.862	0.755	0.844
lymphography	**0.825**	0.548	0.722	0.545	0.685
spect	**0.865**	0.704	0.816	0.726	0.813
tic-tac-toe	**0.837**	0.781	0.815	0.754	0.826
vote	**0.993**	0.96	0.973	0.963	0.976
Australian	**0.878**	0.788	0.837	0.760	0.843
waveform	**0.815**	0.765	0.800	0.774	0.798
Vertebral	**0.890**	0.838	0.877	0.845	0.883
dermatology	**1**	0.967	0.988	0.951	0.991
Satellite	**0.930**	0.905	0.925	0.905	0.921

Table 5
ISSA comparison with other optimization algorithms based on average-number-of-selected-features in 10 runs.

Dataset	ISSA	GOA	PSO	ALO	GA
exactly	**6.4**	7.1	6.7	6.9	6.9
exactly2	**6.2**	6.5	6.6	6.7	7.3
m-of-n	**6.5**	6.6	7	7.1	7.6
sonar	**16**	29.6	26.9	29.3	22.6
ionosphere	**10.7**	15.7	15.1	16.4	11.7
credit	**8.8**	10	9.7	10	9.3
liver	**2.8**	3	3.1	2.9	2.9
spambase	**27.4**	28.7	29.1	29.5	29.2
heart	**4.6**	6.4	6.2	5.9	5
lymphography	**7.4**	8.2	8.5	8.7	8
spect	**9.6**	10.8	10	11.2	10.5
tic-tac-toe	5.5	5.2	**4.8**	**4.8**	5.6
vote	5.1	7.7	7	7.6	**3.5**
Australian	**4.7**	5.9	6.9	6.7	5.2
waveform	12.3	**11.3**	11.5	11.6	13.1
Vertebral	**3**	3.3	3.2	3.4	3.7
dermatology	**11.2**	16.8	15.9	16.5	13
Satellite	18.1	18.7	19.3	**18**	22

Table 6
ISSA Comparison with other optimization algorithms based on average-fitness value in 10 runs.

Dataset	ISSA	GOA	PSO	ALO	GA
exactly	**0.004**	0.275	0.129	0.292	0.095
exactly2	**0.218**	0.259	0.223	0.284	0.247
m-of-n	**0.005**	0.157	0.053	0.173	0.026
sonar	**0.011**	0.099	0.036	0.100	0.027
ionosphere	**0.007**	0.074	0.029	0.066	0.037
credit	**0.204**	0.287	0.248	0.311	0.238
liver	**0.262**	0.298	0.269	0.333	0.286
spambase	**0.053**	0.097	0.066	0.099	0.069
heart	**0.120**	0.213	0.139	0.246	0.157
lymphography	**0.176**	0.451	0.278	0.454	0.315
spect	**0.137**	0.297	0.185	0.275	0.189
tic-tac-toe	**0.167**	0.222	0.188	0.248	0.177
vote	**0.009**	0.044	0.029	0.040	0.025
Australian	**0.123**	0.213	0.165	0.241	0.158
waveform	**0.189**	0.238	0.203	0.229	0.206
Vertebral	**0.113**	0.165	0.127	0.158	0.121
dermatology	**0.002**	0.037	0.014	0.053	0.011
Satellite	**0.074**	0.099	0.079	0.098	0.083

ISSA outperformed all other algorithms in terms of fitness value because it has the minimal classification error among all algorithms. This ISSA superiority resulted from its capability to balance the search process between exploration and exploitation over ISSA iterations. Moreover, based on Fig. 4, ISSA outperformed all other algorithms in convergence over all 18 datasets. Thus, the comparison of ISSA convergence to other algorithms proved its superior performance. In addition, as shown from convergence curves in Fig. 4, ISSA solved the premature convergence in comparison with other algorithm as it balances between exploitation and exploration, and improved population diversity. This stability in ISSA performance resulted from the two improvements into original SSA algorithm which include using OBL strategy and LSA algorithm.

Based on Fig. 4, ISSA has better exploration and exploitation ability than other optimization algorithms. This demonstrates the ability of ISSA to maintain solutions diversity better than other algorithms. Furthermore, the obtained fitness values by ISSA as shown in Table 6 are always superior to other algorithms which confirmed its ability to avoid local optima. Whereas, the other algorithms may easily fall in local optima. In addition, based on the selected number of features by ISSA as shown in Table 5, ISSA has better exploration ability than other algorithms which confirmed

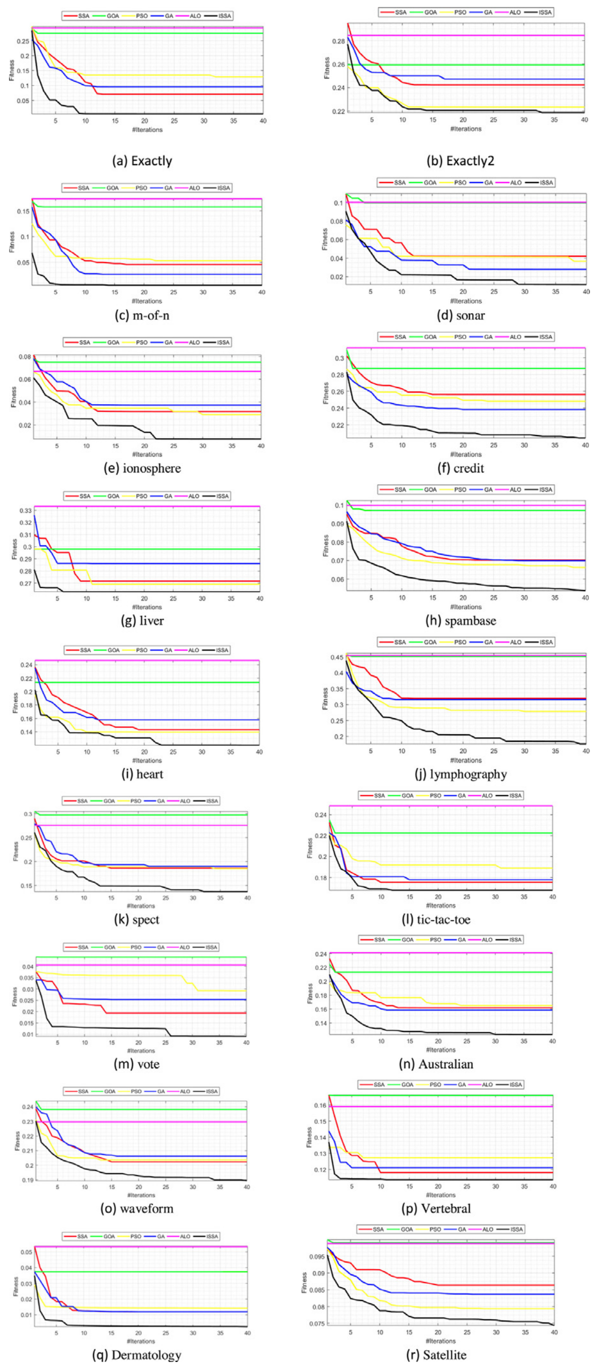

Fig. 4. Convergence of ISSA in comparison to other algorithms.

by selecting less features over 14 datasets out of 18 datasets. This exploration superiority of ISSA was demonstrated by its ability to select fewer number of features and its superiority in comparison with other optimization algorithms based on the accuracy obtained results over 18 datasets.

5. Conclusion

In this work, we proposed an improved variant of SSA algorithm using OBL strategy and LSA algorithm for the feature selection problems. The two improvements, which were embedded into the standard SSA, are used to improve its population diversity and to avoid SSA from trapping into local optima. The ISSA utilizes the benefits of OBL strategy to improve the diversity of SSA population. Furthermore, ISSA was embedded with the new LSA algorithm to avoid it from trapping in local optima. The proposed ISSA algorithm was tested and evaluated over 18 datasets from UCI repository. ISSA algorithm was compared with number of well-known optimization algorithms (SSA, GA, PSO, GOA, and ALO) based on number of factors. These factors include classification accuracy, fitness, and number of selected features. The comparative experiments and evaluations revealed the superiority of ISSA in comparison with other algorithms. In addition, the results affirmed the ability of OBL and LSA in improving the performance and the quality of solutions in SSA algorithm, and fast convergence to find the optimal solution.

There is an important practical implication of the proposed ISSA wrapper feature selection mode. ISSA can be applied to other domains such as data mining, data science, sentiment analysis, medical application, engineering application, and many more. This is based on the obtained results from the conducted experiments on various types of datasets.

One limitation of the proposed ISSA is the selection of more features than other optimization algorithms over four datasets from the 18 used datasets. Therefore, new selection strategy can be used to strengthen the proposed algorithm to select fewer features, and need further research and can be considered as a possible future work.

For future works, the application of ISSA algorithm to real world problems can be investigated such as IDS, cancer detection, and sentiment analysis. In addition, ISSA can be investigated on different types of datasets. Furthermore, we will investigate the combination of the new LSA algorithm with other optimization algorithms. The performance of ISSA can be investigated by employing different classifiers such as neural networks and support vector machines (SVM).

Declaration of Competing Interest

None.

Credit authorship contribution statement

Mohammad Tubishat: Conceptualization, Data curation, Resources, Investigation, Methodology, Writing - original draft, Writing - review & editing. **Norisma Idris:** Funding acquisition, Project administration, Resources, Supervision, Writing - review & editing. **Liyana Shuib:** Validation, Writing - review & editing. **Mohammad A.M. Abushariah:** Resources, Supervision, Writing - review & editing. **Seyedali Mirjalili:** Validation, Formal analysis, Writing - review & editing.

Acknowledgment

This research work is supported by the Research University Grant – Faculty Program (Project No: GPF007D-2018).

References

Acharya, N., & Singh, S. (2018). An IWD-based feature selection method for intrusion detection system. *Soft Computing, 22*(13), 4407–4416.
Adeli, A., & Broumandnia, A. (2018). Image steganalysis using improved particle swarm optimization based feature selection. *Applied Intelligence, 48*(6), 1609–1622.
Ahmad, S. R., Bakar, A. A., & Yaakub, M. R. (2019). Ant colony optimization for text feature selection in sentiment analysis. *Intelligent Data Analysis, 23*(1), 133–158.
Ahmad, S. R., Bakar, A. A., & Yaakub, M. R. (2015). Metaheuristic algorithms for feature selection in sentiment analysis. *Paper presented at the 2015 science and information conference (SAI).*
Alarifi, A., Tolba, A., Al-Makhadmeh, Z., & Said, W. (2018). A big data approach to sentiment analysis using greedy feature selection with cat swarm optimization-based long short-term memory neural networks. *The Journal of Supercomputing*, 1–16.

Aljarah, I., Ala'M, A.-. Z., Faris, H., Hassonah, M. A., Mirjalili, S., & Saadeh, H. (2018). Simultaneous feature selection and support vector machine optimization using the grasshopper optimization algorithm. *Cognitive Computation*, 1–18.

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.

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

Asasi, M. S., Ahanch, M., & Holari, Y. T. (2018). Optimal allocation of distributed generations and shunt capacitors using salp swarm algorithm. *Paper presented at the electrical engineering (ICEE), Iranian conference on*.

Bagheri, M., Nurmanova, V., Abedinia, O., Naderi, M. S., Naderi, M. S., & Ghadimi, N. (2018). A novel wind power forecasting based feature selection and hybrid forecast engine bundled with honey bee mating optimization. *Paper presented at the 2018 IEEE international conference on environment and electrical engineering and 2018 IEEE industrial and commercial power systems Europe (EEEIC/I&CPS Europe)*.

Bairathi, D., & Gopalani, D. (2018). Opposition based salp swarm algorithm for numerical optimization. *Paper presented at the international conference on intelligent systems design and applications*.

Baygi, S. M. H., Karsaz, A., & Elahi, A. (2018). *A hybrid optimal PID-Fuzzy control design for seismic exited structural system against earthquake: A salp swarm algorithm*. *Paper presented at the fuzzy and intelligent systems (CFIS), 2018 6th Iranian joint congress on*.

Beyranvand, P., Kucuktezcan, C. F., Cataltepe, Z., & Genc, V. M. I. (2018). A novel feature selection method for the dynamic security assessment of power systems based on multi-layer perceptrons. *International Journal of Intelligent Systems and Applications in Engineering, 6*(1), 53–58.

Chen, Z., Han, X., Fan, C., Zheng, T., & Mei, S. (2019). A two-stage feature selection method for power system transient stability status prediction. *Energies, 12*(4), 689.

Chen, Z., Tondi, B., Li, X., Ni, R., Zhao, Y., & Barni, M. (2019). Secure detection of image manipulation by means of random feature selection. *IEEE Transactions on Information Forensics and Security, 14*(9), 2454–2469.

Cheng, M.-M., Liu, Y., Hou, Q., Bian, J., Torr, P., Hu, S.-M., et al. (2016). *HFS: Hierarchical feature selection for efficient image segmentation*. *Paper presented at the European conference on computer vision*.

Cheruku, R., Edla, D. R., Kuppili, V., & Dharavath, R. (2018). Rst-batminer: A fuzzy rule miner integrating rough set feature selection and bat optimization for detection of diabetes disease. *Applied Soft Computing, 67*, 764–780.

Damodaran, B. B., Courty, N., & Lefèvre, S. (2017). Sparse Hilbert Schmidt independence criterion and surrogate-kernel-based feature selection for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing, 55*(4), 2385–2398.

Dua, D., & Taniskidou, E. K. (2017). *UCI machine learning repository*: 144. Irvine, CA: University of California, School of Information and Computer Science http://archive.ics.uci.edu/ml.

Dutta, S., Ghatak, S., Das, A. K., Gupta, M., & Dasgupta, S. (2019). Feature selection-based clustering on micro-blogging data. In *Computational intelligence in data mining* (pp. 885–895). Springer.

Eesa, A. S., Orman, Z., & Brifcani, A. M. A. (2015). A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. *Expert Systems with Applications, 42*(5), 2670–2679.

Ekinci, S., & Hekimoglu, B. (2018). Parameter optimization of power system stabilizer via Salp Swarm algorithm. *Paper presented at the 2018 5th international conference on electrical and electronic engineering (ICEEE)*.

El-Fergany, A. A. (2018). Extracting optimal parameters of PEM fuel cells using Salp Swarm Optimizer. *Renewable Energy, 119*, 641–648.

Emary, E., & Zawbaa, H. M. (2019). Feature selection via Lèvy Antlion optimization. *Pattern Analysis and Applications, 22*(3), 857–876.

Emary, E., Zawbaa, H. M., & Hassanien, A. E. (2016). Binary grey wolf optimization approaches for feature selection. *Neurocomputing, 172*, 371–381.

Ewees, A. A., Elaziz, M. A., & Houssein, E. H. (2018). Improved grasshopper optimization algorithm using opposition-based learning. *Expert Systems with Applications, 112*, 156–172.

Faris, H., Mafarja, M. M., Heidari, A. A., Aljarah, I., Ala'M, A.-. Z., Mirjalili, S., et al. (2018). An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. *Knowledge-Based Systems, 154*, 43–67.

Ghadimi, N., Akbarimajd, A., Shayeghi, H., & Abedinia, O. (2019). Application of a new hybrid forecast engine with feature selection algorithm in a power system. *International Journal of Ambient Energy, 40*(5), 494–503.

González, S., Castillo, D., Galvez, J. M., Rojas, I., & Herrera, L. J. (2019). Feature selection and assessment of lung cancer sub-types by applying predictive models. *Paper presented at the international work-conference on artificial neural networks*.

Gu, S., Cheng, R., & Jin, Y. (2018). Feature selection for high-dimensional classification using a competitive swarm optimizer. *Soft Computing, 22*(3), 811–822.

Gupta, D., Julka, A., Jain, S., Aggarwal, T., Khanna, A., Arunkumar, N., et al. (2018). Optimized cuttlefish algorithm for diagnosis of Parkinson's disease. *Cognitive systems research, 52*, 36–48.

Guru, D., Ali, M., & Suhil, M. (2019). A novel feature selection technique for text classification. In *Emerging technologies in data mining and information security* (pp. 721–733). Springer.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*(Mar), 1157–1182.

Hancer, E., Xue, B., & Zhang, M. (2018). Differential evolution for filter feature selection based on information theory and feature ranking. *Knowledge-Based Systems, 140*, 103–119.

Hasanloei, M. A. V., Sheikhpour, R., Sarram, M. A., Sheikhpour, E., & Sharifi, H. (2018). A combined Fisher and Laplacian score for feature selection in QSAR based drug design using compounds with known and unknown activities. *Journal of Computer-Aided Molecular Design, 32*(2), 375–384.

Hegazy, A. E., Makhlouf, M., & El-Tawel, G. S. (2019). Feature selection using chaotic salp swarm algorithm for data classification. *Arabian Journal for Science and Engineering, 44*(4), 3801–3816.

Ibrahim, A., Ahmed, A., Hussein, S., & Hassanien, A. E. (2018). Fish image segmentation using Salp Swarm Algorithm. *Paper presented at the International Conference On Advanced Machine Learning Technologies And Applications*.

Ibrahim, H. T., Mazher, W. J., Ucan, O. N., & Bayat, O. (2017). Feature selection using Salp Swarm algorithm for real biomedical datasets. *IJCSNS, 17*(12), 13.

Jain, I., Jain, V. K., & Jain, R. (2018). Correlation feature selection based improved-binary particle swarm optimization for gene selection and cancer classification. *Applied Soft Computing, 62*, 203–215.

Jayaraman, V., & Sultana, H. P. (2019). Artificial gravitational cuckoo search algorithm along with particle bee optimized associative memory neural network for feature selection in heart disease classification. *Journal of Ambient Intelligence and Humanized Computing*, 1–10.

Jensen, R., & Shen, Q. (2008). *Computational intelligence and feature selection: Rough and fuzzy approaches*: Vol. 8. John Wiley & Sons.

Jothi, G. (2016). Hybrid tolerance rough set–firefly based supervised feature selection for MRI brain tumor image classification. *Applied Soft Computing, 46*, 639–651.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1–2), 273–324.

Kumar, A., Jaiswal, A., Garg, S., Verma, S., & Kumar, S. (2019). Sentiment analysis using cuckoo search for optimized feature selection on Kaggle tweets. *International Journal of Information Retrieval Research (IJIRR), 9*(1), 1–15.

Kuo, R.-. J., Huang, S. L., Zulvia, F. E., & Liao, T. W. (2018). Artificial bee colony-based support vector machines with feature selection and parameter optimization for rule extraction. *Knowledge and Information Systems, 55*(1), 253–274.

Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering, 17*(4), 491–502.

Mafarja, M., & Mirjalili, S. (2018). Whale optimization approaches for wrapper feature selection. *Applied Soft Computing, 62*, 441–453.

Marie-Sainte, S. L., & Alalyani, N. (2018). Firefly algorithm based feature selection for arabic text classification. *Journal of King Saud University-Computer and Information Sciences*.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software, 114*, 163–191.

Mirzaei, S., El Yacoubi, M., Garcia-Salicetti, S., Boudy, J., Kahindo, C., Cristancho-Lacroix, V., et al. (2018). Two-stage feature selection of voice parameters for early Alzheimer's disease prediction. *IRBM, 39*(6), 430–435.

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.

Pratiwi, A. I. (2018). On the feature selection and classification based on information gain for document sentiment analysis. *Applied Computational Intelligence and Soft Computing, 2018*.

Qiu, C. (2019). A novel multi-swarm particle swarm optimization for feature selection. *Genetic Programming and Evolvable Machines*, 1–27.

Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. (2008). Opposition-based differential evolution. *IEEE Transactions on Evolutionary Computation, 12*(1), 64–79.

Rani, M. J., & Devaraj, D. (2019). Two-Stage hybrid gene selection using mutual information and genetic algorithm for cancer data classification. *Journal of Medical Systems, 43*(8), 235.

Ranjbar, S., & Jamali, S. (2019). Fault detection in microgrids using combined classification algorithms and feature selection methods. *Paper presented at the 2019 international conference on protection and automation of power system (IPAPS)*.

Rodrigues, D., Pereira, L. A., Nakamura, R. Y., Costa, K. A., Yang, X.-. S., & Souza, A. N. (2014). A wrapper approach for feature selection based on bat algorithm and optimum-path forest. *Expert Systems with Applications, 41*(5), 2250–2258.

Sapre, S., & Mini, S. (2018). Opposition-based moth flame optimization with Cauchy mutation and evolutionary boundary constraint handling for global optimization. *Soft Computing*, 1–19.

Sarkhel, R., Chowdhury, T. M., Das, M., Das, N., & Nasipuri, M. (2017). A novel Harmony Search algorithm embedded with metaheuristic opposition based learning. *Journal of Intelligent & Fuzzy Systems, 32*(4), 3189–3199.

Sawhney, R., Mathur, P., & Shankar, R. (2018). A firefly algorithm based wrapper-penalty feature selection method for cancer diagnosis. *Paper presented at the international conference on computational science and its applications*.

Sayed, G. I., Hassanien, A. E., & Azar, A. T. (2017). Feature selection via a novel chaotic crow search algorithm. *Neural Computing and Applications*, 1–18.

Sayed, S., Nassef, M., Badr, A., & Farag, I. (2019). A nested genetic algorithm for feature selection in high-dimensional cancer Microarray datasets. *Expert Systems with Applications, 121*, 233–243.

Selvakumar, B., & Muneeswaran, K. (2019). Firefly algorithm based feature selection for network intrusion detection. *Computers & Security, 81*, 148–155.

Selvakumar, K., Karuppiah, M., SaiRamesh, L., Islam, S. H., Hassan, M. M., Fortino, G., et al. (2019). Intelligent temporal classification and fuzzy rough set-based feature selection algorithm for intrusion detection system in WSNs. *Information Sciences, 497*, 77–90.

Shan, X., Liu, K., & Sun, P.-. L. (2016). Modified bat algorithm based on lévy flight and opposition based learning. *Scientific Programming, 2016*.

Sharif, M., Khan, M. A., Iqbal, Z., Azam, M. F., Lali, M. I. U., & Javed, M. Y. (2018). Detection and classification of citrus diseases in agriculture based on optimized weighted segmentation and feature selection. *Computers and Electronics in Agriculture, 150*, 220–234.

Sharma, P., Sundaram, S., Sharma, M., Sharma, A., & Gupta, D. (2019). Diagnosis of Parkinson's disease using modified grey wolf optimization. *Cognitive systems research, 54*, 100–115.

Tolba, M., Rezk, H., Diab, A., & Al-Dhaifallah, M. (2018). A novel robust methodology based salp swarm algorithm for allocation and capacity of renewable distributed generators on distribution grids. *Energies, 11*(10), 2556.

Tubishat, M., Abushariah, M. A., Idris, N., & Aljarah, I. (2018). Improved whale optimization algorithm for feature selection in Arabic sentiment analysis. *Applied Intelligence*, 1–20.

Tubishat, M., Abushariah, M. A., Idris, N., & Aljarah, I. (2019). Improved whale optimization algorithm for feature selection in Arabic sentiment analysis. *Applied Intelligence, 49*(5), 1688–1707.

Turabieh, H., Mafarja, M., & Li, X. (2019). Iterated feature selection algorithms with layered recurrent neural network for software fault prediction. *Expert Systems with Applications, 122*, 27–42.

Verma, O. P., Aggarwal, D., & Patodi, T. (2016). Opposition and dimensional based modified firefly algorithm. *Expert Systems with Applications, 44*, 168–176.

Vijayanand, R., Devaraj, D., & Kannapiran, B. (2018). Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. *Computers & Security, 77*, 304–314.

Wang, M., Wu, C., Wang, L., Xiang, D., & Huang, X. (2019). A feature selection approach for hyperspectral image based on modified ant lion optimizer. *Knowledge-Based Systems, 168*, 39–48.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82.

Yang, J., Liu, S., Tao, W., & Hu, C. (2018). A distributed security feature selection algorithm based on K-means in power grid system. *Paper presented at the international conference on cloud computing and security*.

Yao, C., Liu, Y.-F., Jiang, B., Han, J., & Han, J. (2017). LLE score: A new filter-based unsupervised feature selection method based on nonlinear manifold embedding and its application to image recognition. *IEEE Transactions on Image Processing, 26*(11), 5257–5269.

Zhang, D., Zou, L., Zhou, X., & He, F. (2018). Integrating feature selection and feature extraction methods with deep learning to predict clinical outcome of breast cancer. *IEEE Access, 6*, 28936–28944.

Zhang, J., Wang, Z., & Luo, X. (2018). Parameter estimation for soil water retention curve using the Salp Swarm Algorithm. *Water, 10*(6), 815.

Zhang, L., Mistry, K., Lim, C. P., & Neoh, S. C. (2018). Feature selection using firefly optimization for classification and regression models. *Decision Support Systems, 106*, 64–85.

Zhou, J., Fang, W., Wu, X., Sun, J., & Cheng, S. (2016). An opposition-based learning competitive particle swarm optimizer. *Paper presented at the evolutionary computation (CEC), 2016 IEEE Congress on*.

Zhou, Y., Hao, J.-. K., & Duval, B. (2017). Opposition-based memetic search for the maximum diversity problem. *IEEE Transactions on Evolutionary Computation, 21*(5), 731–745.