Pattern Analysis and Applications (2019) 22:857–876
https://doi.org/10.1007/s10044-018-0695-2

THEORETICAL ADVANCES

Feature selection via Lèvy Antlion optimization

E. Emary[1] · Hossam M. Zawbaa[2,3]

Received: 17 September 2016 / Accepted: 9 March 2018 / Published online: 19 March 2018

Abstract
In this paper, a modification of the newly proposed antlion optimization (ALO) is introduced and applied to feature selection relied on the Lèvy flights. ALO method is one of the encouraging swarm intelligence algorithms which make use of random walking to perform the exploration and exploitation operations. Random walks based on uniform distribution is responsible for premature convergence and stagnation. A Lèvy flight random walk is suggested as a permutation for performing a local search. Lèvy random walking grants the optimization ability to generate several solutions that are apart from existing solutions and furthermore enables it to escape from local minima and much efficient in examining large search area. The proposed Lèvy antlion optimization (LALO) algorithm is applied in a wrapper-based mode to select optimal feature combination that maximizing classification accuracy while minimizing the number of selected features. LALO algorithm is applied on 21 different benchmark datasets against genetic algorithm (GA), particle swarm optimization (PSO), and the native ALO methods. Different initialization methods and several evaluation criteria are employed to assess algorithm diversification and intensification of the optimization algorithms. The experimental results demonstrate the significant improvement in the proposed LALO over the native ALO and many well-known methods used in feature selection.

Keywords Lèvy Antlion optimization · Lèvy flight · Feature selection · Bio-inspired optimization

1 Introduction

Each dataset regularly has an enormous number of features, especially in pattern recognition and classification applications. The primary goal of feature selection is a better understanding of the underlying method which generated the data to extract a subset of relevant features from a huge number of available features [1]. The selected feature set will improve the classifier performance and provide a faster and more cost-effective classification that leads to obtaining comparable or even better classification performance from using the full features [2].

✉ Hossam M. Zawbaa
hossam.zawbaa@gmail.com

E. Emary
eidemary@yahoo.com

1 Faculty of Computers and Information, Cairo University, Giza, Egypt

2 Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

3 Faculty of Computers and Information, Beni-Suef University, Beni-Suef, Egypt

The feature selection is recognized as a multi-objective problem that minimizes the size of selected features and maximizes the classification performance. Both objectives are contradicted, and the optimal solution requires to be performed in the appearance of a tradeoff between them. The feature selection methods split into two primary approaches: filter-based and wrapper-based [3]. The filter-based method is not dependent on machine learning technique, and they are participated to be computationally inexpensive. Furthermore, filter-based methods examine the search space for a feature set that optimizing given data-dependent criteria rather than classification-dependent criteria [4].

The wrapper-based method includes a machine learning technique as part of the evaluation function which allows the wrapper-based to obtain better results than filter-based method [5]. The size of search space is exponentially increased concerning the number of features in a given dataset [1]. Hence, in practice, the exhaustive search methods are impossible to get the optimal solution in the most cases. The diversity of search methods are employed to solve feature selection problems like greedy search based on sequential forward selection (SFS) [6] and sequential backward selection (SBS) [7]. However, these classical optimization methods have some restriction in solving the optimization problems. So that, evolutionary computation (EC) algorithms are the option for addressing these shortcomings and their global search ability [8, 9].

Various heuristic methods simulate the behavior of biological and physical systems in nature, and it has been proposed as powerful techniques for global optimizations [10]. EC algorithms are motivated by nature, social behavior, and biological behavior (of animals, birds, fish, bat, Firefly, wolves, spiders, antlion, etc.) in a pack [11]. Many researchers have suggested various computational methods which imitate the behavior of these kinds for investigating their optimal solution [12–15]. EC algorithms were hired to search adaptively the search area for an optimal subset using a set of search agents who communicate in a social manner to reach a global solution in the minimum number of evaluations [16].

In EC algorithms, it is necessary to have the proper balance between exploration and exploitation [17]. Genetic algorithm (GA) and particle swarm optimization (PSO) are the most common evolutionary computation methods [18]. GA was the first evolutionary-based algorithm introduced in the literature and developed based on the natural process of evolution through reproduction [19]. A feature selection algorithm based on GA using the fuzzy set as fitness function [20]. A hybrid wrapper and filter feature selection with a memetic framework for the classification problem. This method integrated filter ranking with GA to maximize the classification accuracy and selected the feature subsets [21]. However, PSO with the same fitness function achieves better performance than GA algorithm in [22]. A feature selection strategy using rough-set and PSO is introduced in [23]. Rough-set has been applied as a feature selection method to find the optimally selected features, and PSO will discover the optimal feature subset.

The ant colony optimization (ACO) method based on wrapper feature selection algorithm applied in network intrusion detection and used Fisher discrimination rate to adopt the heuristic information [24]. In the artificial bee colony (ABC) algorithm, several behaviors of the bees provide the opportunity to build robust balancing method between exploration and exploitation [25]. In ABC, the employer bees try to find a food source and advertise the other bees [26]. The onlooker bees follow their interesting employer, and the scout bee flies spontaneously to find the best food source [27]. A virtual bee algorithm (VBA) is applied to optimize the numerical function in 2-D using a swarm of virtual bees, which move randomly in the search space and interact to find food sources. The interactions between these virtual bees result in the possible solution for the optimization problem [28]. A proposed approach based on the honeybees natural behavior that randomly produced worker bees are moved in the direction of the elite bee. The elite bee describes the optimal (near to optimal) solution [29].

Artificial fish swarm (AFS) algorithm mimics the stimulant reaction by controlling the tail and fin. AFS is a robust stochastic technique based on the fish movement and its intelligence during the food finding process [30]. Antlion optimization (ALO) algorithm [31, 32] is a relatively new EC method which is computationally less costly than other EC techniques. The proposed simulated annealing (SA) is employed to optimize the feature selection subset in marketing applications for designating large-scale linear regression model [33].

Lately, many types of research have studied the animals and insects flight behavior that represented the general characteristics of Lèvy flights [34]. The fruit flies examine their landscape doing a series of straight flight routes with the 90° turn, leading to a Lèvy flight interrupted scale-free search pattern. Therefore, Lèvy flight behavior has been employed to optimization and optimal search, and preliminary results show its promising capability [35]. Lèvy flight process represents the optimum random search pattern and is frequently detected in nature [36]. Broadly speaking, the Lèvy flight is a random walk whose step length is pulled from the Lèvy distribution [37].

The aggregate aim of this paper is to propose a variant of ALO which exploits Lèvy flight for performing a local and global search. The proposed Lèvy flight version of ALO is applied for feature selection approach, which selects the minimal number of features and obtains comparable or even best classification efficiency from using all attributes and conventional feature selection techniques. The rest of this paper is organized as the follows: Sect. 2 provides the background information of antlion optimization algorithm (ALO) and Lèvy flights. Section 3 describes the proposed antlion optimization algorithm (ALO)-based multi-objective feature selection algorithms. The result of experiments with discussions is presented in Sect. 5. Finally, conclusions and future work are provided in Sect. 6.

2 Antlion optimization (ALO)

Antlion optimization (ALO) is a newly proposed bio-inspired optimization method that was suggested by Mirjalili [32]. ALO algorithm mimics the antlions hunting behavior in nature. The following two subsections discuss the inspiration and operators of the artificial algorithm [38].

Antlions (doodlebugs) belong to the Myrmeleontidae family and Neuroptera order [32]. An antlion larva digs a cone-shaped hole in the sand. Once caught, the prey tries to stampede from the trap, and the antlion tries to catch its prey. Antlions intelligently throw sands toward to edge of the hole to *slide* the victim into the bottom of the hole. By consuming the prey, the antlion prepares the hole for the next hunt. The trap size for an antlion is affected by the level of hunger and the moon shape. Antlions tend to dig out larger traps as they become hungrier and when the moon is full. They have been developed and acclimated to this way to increase their chance of survival [39].

Based on the above description of antlions, the ALO is composed of two sets of agents, namely ant and antlions. Antlions are the hunters and are selected as the fittest agents and never change their location unless they replace a given ant. Ants perform random walks in the search space and may be consumed by an antlion if such antlion traps it. The ant's position is updated according to the formula in Eq. (1):

$$\text{Ant}_i^t = \frac{R_A^t + R_E^t}{2}, \tag{1}$$

where R_A^t is the position of randomly walking ant indexed i; called Ant_i, around a roulette wheel selected antlion indexed A and R_E^t is the position of randomly walking ant indexed i, called Ant_i, around the elite antlion indexed E in the ant population. Antlions fitness is used by the roulette wheel operator to select an antlion with index A to perform the ant random walk around it while the elite antlion is established as the fittest antlion with index E.

The random walking of an Ant_i^t around a given Antlion_j^t is formulated as in Eq. (2):

$$R_j^t = \frac{(X_i - a_i) \times (d_i - c_i^t)}{(b_i^t - a_i)} + c_i, \tag{2}$$

where R_j^t is the position of ant i after performing random walk around antlion j at iteration t, a_i is the minimum step of random walk X_i^t in i-th dimension, and b_i is the maximum step of random walk X_i^t in i-th dimension, X_i is defined in Eq. (3), c, d are the lower and upper bounds of the random walk. Equation (2) is directly used to define R_A^t and R_E^t of Eq. (1).

$$\begin{aligned} X(t) &= [0, \text{cumsum}(2r(t_1) - 1); \text{cumsum}(2r(t_2) - 1);\\ &\quad \ldots ;\text{cumsum}(2r(t_T) - 1)], \end{aligned} \tag{3}$$

where cumsum calculates the cumulative sum and models the accumulation of successive random steps forming a random walk until time t, and $r(t)$ is a stochastic function characterized as in Eq. (4):

$$r(t) = \begin{cases} 1 & \text{if rand} > 0.5 \\ 0 & \text{if rand} \leq 0.5, \end{cases} \tag{4}$$

where rand is a random number relied on uniform distribution. c, d parameters are adapted according to Eqs. (5) and (6) to limit the range of the random walk around the given antlion.

$$c_i^t = \begin{cases} \frac{lb}{I} + X_{\text{Antlion}_j}^t & \text{if rand} < 0.5 \\ \frac{-lb}{I} + X_{\text{Antlion}_j}^t & \text{otherwise,} \end{cases} \tag{5}$$

$$d_i^t = \begin{cases} \frac{ub}{I} + X_{\text{Antlion}_j}^t & \text{if rand} > 0.5 \\ \frac{-ub}{I} + X_{\text{Antlion}_j}^t & \text{otherwise,} \end{cases} \tag{6}$$

where lb and ub are the lower limit and upper limit for dimension i, and I is a parameter that monitors the exploration/exploitation rate and is defined as in Eq. (7):

$$I = 10^w \frac{t}{T}, \tag{7}$$

where T is the maximum number of iterations, w is a constant depicted the current iteration ($w = 2$ when $t > 0.1T$, $w = 3$ when $t > 0.5T$, $w = 4$ when $t > 0.75T$, $w = 5$ when $t > 0.9T$, and $w = 6$ when $t > 0.95T$). The constant w may update the accuracy level of exploitation.

Finally, the selection operation is applied where an antlion is replaced by an ant if the ant becomes fitter. According to the above formulation, the antlion optimization can be described in the following algorithm (1) [40].

```
Algorithm 1: Antlion optimization (ALO) algorithm
Input: Search space, fitness function, number of ants and antlions, number of
       iterations ($T$)
Output: The elitist antlion and its fitness
 1. Initialize a population of $n$ ants' positions and $n$ antlions' positions randomly.
 2. Calculate the fitness of all ants and antlions.
 3. Find the Elite antlion $E$; fittest.
 4. $t$=0.
 5. while (Stopping condition not fit)
    foreach $Ant_i$ do
        – Select an antlion $A$ using roulette wheel.
        – Perform random walk of $Ant_i$ around $Antlion_A$ using equation (2); $R_A^t$.
        – Perform random walk of $Ant_i$ around $Antlion_E$ using equation (2); $R_E^t$.
        – Update the position of $Ant_i$ using equation (1).
    end
 6. Recalculate the fitness of all ants.
 7. Merge the ants and antlions and sort all according to fitness and select the best $n$
    agents as the next antlions and the worst $n$ as the ants(Catching Prey).
 8. Update elite if an antlion becomes fitter than the elite.
 9. End while
```

ALO provides very competitive results in terms of improved exploration, local optima avoidance, exploitation, and convergence in diverse shapes of optimization functions and hence found in solving different problems in different domains [32].

3 The proposed Lèvy antlion optimization (LALO)

Randomization has a significant role in both *exploration* and *exploitation*, or *diversification* and *intensification* and the essence of such randomization is the random walk [41]. A *random walk* is a random process that consists of taking a

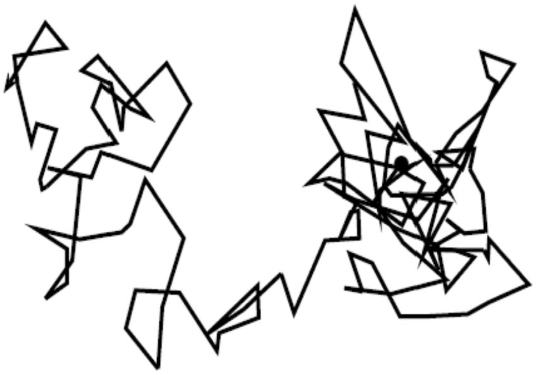

Fig. 1 Brownian motion with a gaussian step-size distribution and the path of 50 steps starting at the origin; after [41]

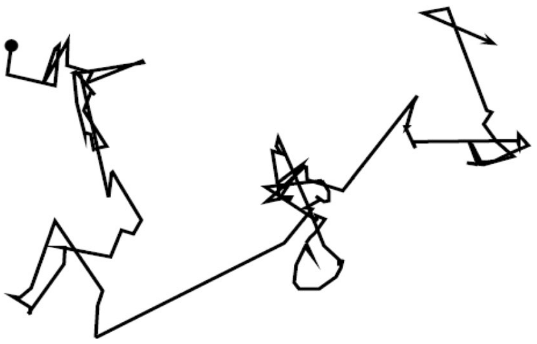

Fig. 2 Lèvy flights in consecutive 50 steps starting at the origin; after [41]

series of consecutive random steps. The random walk may be written as a sum of consecutive random steps as in Eq. (8).

$$S_N = \sum_{i=1}^{N} X_i, \tag{8}$$

where X_i is a random step drawn from a random distribution.

If the step length obeys the Gaussian distribution, the random walk becomes the Brownian motion, as shown in Fig. 1 [41]. Far-field randomization should generate a fraction of solutions and be far enough from the current best solution. Such far solutions help a given optimizer to escape from local optima and avoid stagnation [42]. A variant of Brownian motion that uses Lèvy is expected to reach the end. Lèvy distribution is a heavy-tailed distribution that has infinite variance.

Mathematically speaking, a simple version of Lèvy distribution can be defined as in Eq. (9):

$$L(s,\gamma,\mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}}\exp\left(-\frac{\gamma}{2(s-\mu)}\right)\frac{1}{(s-\mu)^{3/2}}, & 0 < \mu < s < \infty \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where μ is minimum step size, and γ is a scale parameter. As $s \rightarrow \infty$, we have the simple form of Lèvy as in Eq. (10). Figure 2 shows a sample random walking using Lèvy distribution.

$$L(s,\gamma,\mu) = \sqrt{\frac{\gamma}{2\pi}}\frac{1}{(s-\mu)^{3/2}} \tag{10}$$

When the step length obeys the Lèvy distribution, such a random walk is called a Lèvy flight or Lèvy walk.

Lèvy flights are more efficient than Brownian random walks in exploring unknown, large-scale search space which can be observed from the large abrupt jumps. Mathematically, this can be interpreted by the fact that the variance increases much faster than the linear relationship of the Brownian random walk. The facts as mentioned earlier about the Lèvy flights random walk motivate exploiting it in performing the random walking of ants.

The updated version of ALO random walk based on Lèvy flights' random walk is performed as in Eq. (11).

$$\mathrm{RW}_i = \mathrm{ant}_i^t + \omega \times \frac{u}{|v|^{\frac{1}{\beta}}} \times r \times \left[\mathrm{antLion}_j^t - \mathrm{ant}_i^t\right] \tag{11}$$

where RW_i is the random walk of ant_i around $\mathrm{antLion}_j$ at time t, ω is a scale parameter controlling the scale of the random walk and is defined according to Eq. (13), v, r are random number drawn from normal distribution $N(0, 1)$, β is constant controlling the distribution skewness and u is set as $r2 * \varphi$ with $r2$ random number drawn from normal distribution $N(0, 1)$ and φ is defined in Eq. (12). The above equation allows an ant to be repositioned following a random walking with Lèvy steps.

$$\varphi = \left(\frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left[\frac{1+\beta}{2}\beta 2^{\frac{\beta-1}{2}}\right]}\right)^{\frac{1}{\beta}} \tag{12}$$

Such implementation of the Lèvy distribution is based on Mantegna algorithm for a symmetric Lèvy stable distribution [43] which is said to be the easiest way.

$$\omega = 1 - \frac{t}{T} \tag{13}$$

where t, T are the current iteration and the maximum number of allowed iterations in order. Such equation allows the optimizer to be more diverse at the begin of optimization and then narrows such scale to allow for exploitation at the end stages of optimization.

Based on the above updated random walk formulation, the position update for ant_i^t is formulated as in Eq. (14).

$$\text{ant}_i^t = \frac{\text{RW}_A^t + \text{RW}_E^t}{2}, \tag{14}$$

where RW_A^t is the random walk of ant_i around a roulette wheel selected antLion_j and such random walk is obtained by applying Eq. (11) and RW_E^t is obtained by applying random walk of ant_i around the elite antlion antLion_k and is calculated using Eq. (11) too.

The Lèvy antlion optimization (LALO) is comprised of fundamental building phases as described in Fig. 3.

σ^2 defined in [44] was employed as an indicator to prove the convergence of an algorithm to premature or global optima; see Eq. (15).

$$\sigma^2 = \sum_{i=1}^{N} \left(\frac{f_i - f_{\text{avg}}}{f} \right)^2 \tag{15}$$

where σ^2 is the population variance indicator, N is the number of search agents, f_i is the fitness of search agent number i, f_{avg} is the current average fitness of the swarm, and f is the normalized calibration factor to confine σ^2 and is defined as in Eq. (16).

$$f = \begin{cases} \max |f_i - f_{\text{avg}}|, & \max |f_i - f_{\text{avg}}| > 1 \\ 1, & \text{otherwise} \end{cases} \tag{16}$$

In this study, this function is used to ensure convergence of the ALO and its proposed variant LALO. Figure 4 depicts the σ^2 for both ALO and LALO averaged for 30 different runs on sample data. We can notice from the figure that the ALO approaches a value of 0 for the indicator σ^2 at an earlier time, but there is no grantee of reaching the global optima and at this stage, the algorithm stagnates and can not further enhance its search agents. On the other hand, we can recognize that LALO keeps, respectively, higher σ^2 for a long time allowing for diversity of the population and hence allowing for additional exploration of the search space even at the next optimization steps.

Figure 5 depicts the convergence performance of the proposed LALO versus the convergence of ALO on the average over 20 runs with random initialization. We can see that the LALO keeps it exploration capability even at later optimization steps thanks to the random walks received by the Lèvy distribution that has infinite variance.

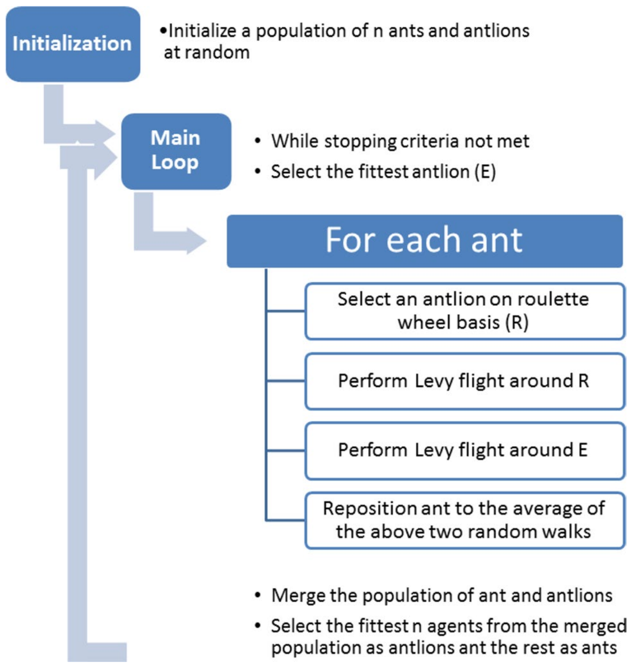
Initialization
•Initialize a population of n ants and antlions at random
Main Loop
• While stopping criteria not met
• Select the fittest antlion (E)
For each ant
Select an antlion on roulette wheel basis (R)
Perform Levy flight around R
Perform Levy flight around E
Reposition ant to the average of the above two random walks
• Merge the population of ant and antlions
• Select the fittest n agents from the merged population as antlions ant the rest as ants

Fig. 3 The proposed Lèvy antlion optimization (LALO) algorithm

4 LALO applied for feature selection

In this study, *wrapper approach* for feature selection is used that means the search area is explored to find a feature subset guided by classification performance. This approach may be slow since the classifier must be retrained on all candidate solutions and its performance must be measured. Therefore, the intelligent search for discovering the search space is necessary. The goals are the maximization of classification performance and the minimization of the selected number of features. Of course, such fitness function can be enhanced to incorporate other classification performance measure such as entropy which is implicitly included in the initialization of search agents in some of our experiments as will be explained in the results section. The used fitness function is minimization as shown in Eq. (17) [45]:

$$\alpha(1 - P) + (1 - \alpha)\left(\frac{N_f}{N_t}\right), \tag{17}$$

where P is the classifier performance measure and N_f, N_t are the size of the selected feature subset and the total number of features in the dataset in order.

$\alpha \in [0, 1]$ define the weights of the sub-goals. P is commonly measured as in Eq. (18):

$$P = \frac{N_c}{N_t}, \tag{18}$$

where N_c, N_t are a number of correctly classified data points and the total number of data points in the dataset, respectively.

The number of dimensions (variables) in the optimization problem is equal to the number of features, and each variable is limited to the range [0, 1]. To decide whether a feature will be selected or not at the evaluation stages, its corresponding variable is threshold using static threshold at 0.5; as shown in Eq. (19):

$$\begin{cases} x_{ij} < 0.5 & \text{Selected feature} \\ x_{ij} \geq 0.5 & \text{Unselected feature} \end{cases} \tag{19}$$

where x_{ij} is the continuous position of search agent i in dimension j. Therefore, an ant/antlion position represents a selected feature set with the positions value increase for dimensions/features that are candidate to be selected for classification.

The proposed LALO for feature selection algorithm is outlined in the algorithm 2.

```
Algorithm 2: LALO algorithm for feature selection
Input: Feature space, fitness function, number of ants, antlions, and number of
       iterations ($T$)
Output: The elitist antlion and its fitness
 1. Initialize a population of $n$ ants' positions and $n$ antlions' positions randomly.
 2. Calculate the fitness of all ants and antlions using equation (17).
 3. Find the fittest antlion (elite).
 4. $t$=0.
 5. while ($t \leq T$)
    foreach $ant_i$ do
        - Select an antlion using Roulette wheel (building trap); namely $Antlion_r$.
        - Perform random walk of $ant_i$ around $Antlion_r$ using Lèvy flight equation (11).
        - Perform random walk of $ant_i$ around Elite Antlion using Lèvy flight equation
          (11).
        - update $ant_i$ position as the average of the above two random walks; see
          equation (1).
    end
 6. Calculate the fitness of all ants using equation (17).
 7. Replace an antlion with its corresponding ant it if becomes fitter (catching prey).
 8. Update elite if an antlion becomes fitter than the elite.
    end while
```

5 Experimental results

5.1 Dataset and parameters setting

Table 1 summarizes the 21 used datasets for further experiments. The datasets are drawn from the UCI data repository [46]. As evident in the table, we selected datasets with *various dimensions* and *several numbers of instances* to assess the performance on different problems with different complexity. Each dataset is divided in *cross-validation* [47] manner for evaluation. In the k-fold cross-validation, $K - 1$ folds are used for training and validation and the last fold is used for testing. This process is repeated M times. Hence, an individual optimizer is evaluated $K * M$ times for individual dataset. In this study, we used $K = 3$ where the data is divided into three equal portions. The *training* part is used to train the used classifier through optimization and at the final evaluation. The *validation* part used to assess the performance of the classifiers at the optimization time, while the *testing* part is used to evaluate the finally selected features given the trained classifier.

Since the primary goal is to assess the performance of the proposed variant of ALO against the native ALO, we applied both optimizers on the data in Table 1 in cross-validation manner with same parameter settings for both algorithms. A list of parameter setting for both optimizers is mentioned in Table 2. As can be noticed from the table, an individual stochastic algorithm is run for 60 runs to assess its mean performance as a stochastic algorithm. Both algorithms use the same number of search agents; ants and antlions and run for the same number of iterations and in the same search space using the same fitness function. The implementation of Lèvy distribution is based on Mantegna algorithm [41] as an easy implementation with single parameter β controlling the skewness of the distribution. A scaling factor *stepSize* is used to control the exploitation rate. This parameter is set to decrease linearly as the optimization progresses in a linear fashion as we mentioned in Eq. (13).

In multi-agent optimizers with stochastic nature, the initial agent's positions affect the performance of the optimizer. Therefore, in this study we evaluated the two algorithms using different settings for the initial agents so that we can assess:

- The capability of the algorithm to generate several solutions.
- The ability of the algorithm to escape from local optima.
- The intensification capability of the algorithm.

Table 1 List of datasets used in the study

Index	Dataset	No. dims	No. instances
1	Abalone	8	4177
2	Audiology	63	224
3	Australian	14	690
4	Breastcancer	9	699
5	CongressEW	16	435
6	Ecoli	7	336
7	Exactly	13	1000
8	Glass	9	214
9	KrvskpEW	36	3196
10	M-of-n	13	1000
11	NbutanolsConc	12	80
12	Page	10	5473
13	Segmentation	19	2310
14	Tic-tac-toe	9	958
15	Vowel	10	528
16	WineEW	13	178
17	Yeast	8	1484
18	Zoo	16	101
19	PenglungEW	325	73
20	Clean2	166	6598
21	Clean1	166	476

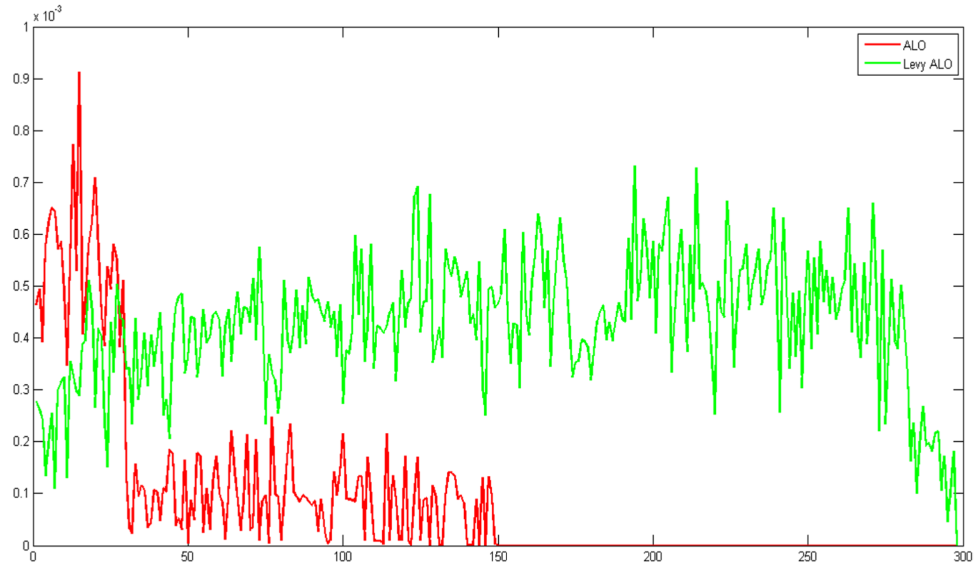

Fig. 4 The evolution curve of the variance of fitness in global convergence of LALO and ALO

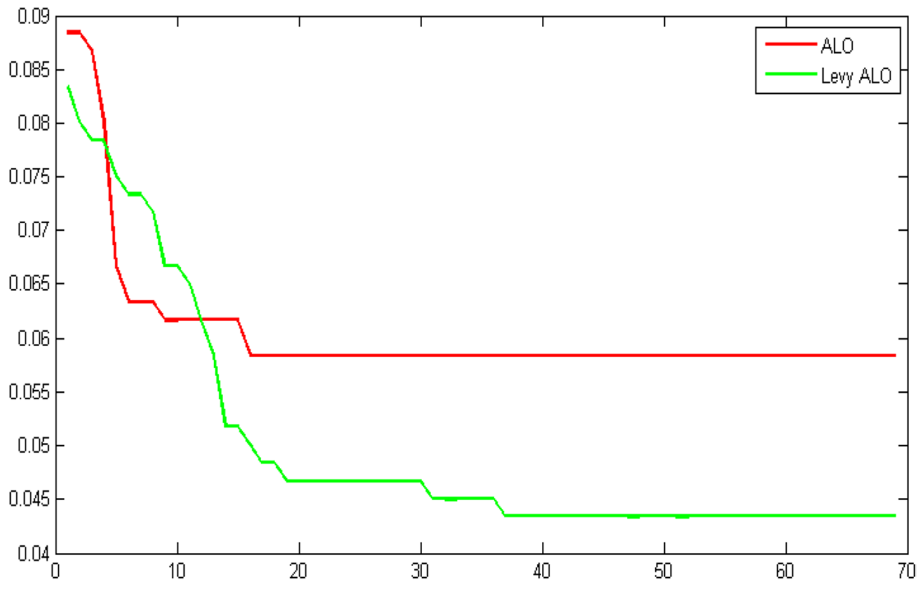

Fig. 5 LALO versus ALO convergence on the average

Table 2 Parameter settings for different optimization methods

Parameter	Value
K for cross-validation	3
M the number of runs	20
No. of search agents	8
No. of iterations	70
Problem dimension	Same as number of features
Search domain	[0 1]
α parameter in the fitness function	0.99
β of Lèvy distribution	$\frac{3}{2}$

To reach such end, we made use of five initializers that forces the optimizers to be initialized with search agents that:

- Are apart from the expected global optima.
- Has very small diversity.
- Close to the expected global optima.
- Contains a near-optimal solution.

Five initialization methods depicted as an example in Fig. 6 and can be detailed as:

1. Uniform initialization: In this method, all the search agents are placed in the search space at random making use of uniform random distribution. That is the most common initialization technique where every point in the search space is a candidate to select as a position for a search agent with the same probability to other points. This initialization method can be formulated as:

$$X_i^d = lb^d + \text{rand}(ub^d - lb^d) \tag{20}$$

with X_i^d is the position of search agent i in dimension d, lb, ub are the lower and upper limits of dimension d and rand is a random number drawn from the uniform distribution.
2. Small initialization: This method initializes the search agents on the terminals of the hypercube of the search space. In the context of feature selection, search agents are initialized with the minimum number of randomly selected features. Therefore, if the number of agents is less than the number of features, we will see that each search agent will have a single dimension with 1. Of course, the optimizer will search for the feature(s) to be set to 1 to enhance the fitness function value as in the standard forward selection of features. This initializer is expected to place the search agents apart from the optimal solution.
3. Large initialization: This method also initializes the search agents on the terminal of the search space hypercube randomly, but it is much biased toward the selection of features rather than the deselection of features. This initializer is expected to initialize the search agents closer to the optimal solution.
4. Mixed initialization: This initializer is a compromise between the small and large initializers where 50% of the search agents are initialized with small initialization and the remaining 50% is initialized with large initialization. This method provides much diversity in the population than the small and large initializers.
5. Minimum redundancy maximum relevance (MRMR) feature criterion combines two optimization criteria [48] in a single formula outlined in Eq. (21).

$$\max_{g_i \in G - S_{m-1}} (I(g_i;c) - \frac{1}{m-1} \sum_{g_j \in S_{m-1}} I(g_i;g_j)) \tag{21}$$

where $(x; y)$ is the mutual information between two random variables X and Y that measures the mutual dependence of these two variables, c is the class labels, and g_i are the values of the variable number i in the data.

In the use of mutual information concept, MRMR method selects variables that have the highest relevance with the target class and are minimally redundant, i.e., selects variables that are maximally dissimilar to each other. The value of each dimension (feature) is set proportional to how to match it can predict the output class label and how much it is distinct from other features. The assessed MRMR measure for individual features is normalized along the feature set as shown in Eq. (22).

$$P_i^d = \frac{\text{MRMR}_i}{\max_{j=1}^{d} \text{MRMR}_j} \tag{22}$$

where P_i is the search agent position in dimension d, MRMR_i is the MRMR value for feature i, and d is the problem dimension.

The obtained position value from the MRMR method is used to initialize one search agent, and the rest of search agents are set at random positions in the search space. This initializer forces one search agent to be very close to the optimal but generally is not the optimal as the criteria used are not directly related to classification performance.

5.2 Evaluation criteria

The well-known K-nearest neighbor (KNN) is used as a classifier to evaluate the final classification performance of each algorithm and wrapper-based feature selection. KNN is employed in the experiments based on trial and error basis where the best choice of K is selected ($K = 5$) as the best performing on all the datasets [49].

A set of evaluation indicators is used to assess different aspects of performance.

Mean fitness is used to evaluate the average performance of the optimizer over all the runs [50].

Standard deviation (std) is a representation of the variation of the obtained best solutions found for running a stochastic optimizer for M different runs. Std is used as an indicator for optimization stability and robustness, whereas Std is smaller; this means that the optimizer always converges to the same solution, while larger values for Std mean much random performance [51].

Average feature size is the average ratio of the selected features to the total number of features which is the secondary objective of the used fitness function. This measure can be formulated as in Eq. (23).

$$\text{Avg}SZ = \frac{1}{M} \sum_{i=1}^{M} \frac{\text{size}(g_*^i)}{D}, \tag{23}$$

where size(x) is the number of on values for the vector x, and D is the number of features in the original dataset.

Table 3 Mean fitness using different initializers for all multi-agent optimizers over various datasets and the fitness value for the deterministic single solution methods

Dataset	Small				Large			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.375	0.378	0.372	**0.369**	0.413	0.416	0.370	**0.369**
2	**0.267**	**0.267**	**0.267**	**0.267**	0.288	0.288	**0.273**	0.288
3	0.234	0.178	**0.146**	0.148	0.173	0.187	**0.157**	0.162
4	0.092	0.076	**0.025**	**0.025**	0.043	0.060	0.025	**0.024**
5	0.061	0.108	**0.042**	0.044	0.086	0.118	0.044	**0.043**
6	0.145	**0.145**	0.148	**0.145**	0.147	0.147	**0.146**	0.147
7	0.332	0.336	**0.286**	0.287	0.339	0.378	0.279	**0.242**
8	**0.328**	**0.328**	0.341	**0.328**	**0.328**	**0.328**	0.331	**0.328**
9	0.066	0.086	**0.050**	0.052	0.121	0.139	**0.051**	**0.051**
10	0.192	0.190	0.106	**0.091**	0.173	0.144	0.103	**0.088**
11	**0.534**	**0.534**	0.540	**0.534**	**0.543**	**0.543**	0.545	**0.543**
12	**0.044**	**0.044**	0.045	**0.044**	**0.044**	**0.044**	**0.044**	**0.044**
13	0.080	0.145	0.061	**0.058**	0.066	0.168	**0.059**	**0.059**
14	0.281	0.318	0.238	**0.234**	0.268	0.342	0.236	**0.231**
15	0.361	0.334	0.313	**0.302**	0.366	0.411	0.313	**0.305**
16	0.028	0.031	0.016	**0.015**	0.086	0.052	0.021	**0.015**
17	0.452	0.452	**0.448**	0.452	**0.445**	**0.445**	0.448	**0.445**
18	0.119	0.117	0.096	**0.089**	0.183	0.104	0.101	**0.085**
19	0.143	**0.130**	0.141	0.143	0.189	0.189	0.152	**0.149**
20	0.056	**0.035**	0.046	0.049	0.058	0.057	0.048	**0.046**
21	0.196	0.221	0.191	**0.186**	0.339	0.339	0.249	**0.222**
Dataset	Mixed				MRMR			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.435	0.457	0.371	**0.370**	0.395	0.481	0.371	**0.369**
2	0.275	0.275	**0.271**	0.275	0.282	0.282	**0.261**	0.282
3	0.183	0.223	**0.159**	0.162	0.176	0.204	0.149	**0.145**
4	0.094	0.065	0.026	**0.024**	0.101	0.075	**0.024**	**0.024**
5	0.071	0.055	**0.042**	0.044	0.061	0.097	0.042	**0.041**
6	**0.148**	**0.148**	**0.148**	**0.148**	**0.146**	**0.146**	0.147	**0.146**
7	0.353	0.376	0.283	**0.251**	0.291	0.321	0.282	**0.248**
8	0.333	**0.330**	0.340	**0.330**	**0.326**	**0.326**	0.331	**0.326**
9	0.077	0.063	0.052	**0.046**	0.112	0.101	0.044	**0.042**
10	0.125	0.135	0.103	**0.080**	0.093	0.042	0.038	**0.037**
11	0.531	**0.531**	0.539	**0.531**	0.540	0.540	**0.534**	0.540
12	0.044	**0.044**	**0.044**	**0.044**	**0.044**	**0.044**	**0.044**	**0.044**
13	0.095	0.114	0.061	**0.059**	0.126	0.118	0.061	**0.059**
14	0.319	0.304	0.241	**0.234**	0.308	0.258	0.240	**0.231**
15	0.376	0.363	0.308	**0.307**	0.368	0.345	0.304	**0.301**
16	0.020	0.124	0.019	**0.015**	0.111	0.089	0.016	**0.015**
17	0.447	**0.447**	**0.447**	**0.447**	0.450	0.450	**0.447**	0.450
18	0.131	0.205	0.092	**0.089**	0.155	0.173	**0.084**	0.089
19	0.137	0.142	**0.132**	0.166	0.141	0.143	**0.138**	**0.138**
20	0.054	0.051	0.049	**0.040**	0.047	0.041	0.046	**0.040**
21	0.212	0.205	0.184	**0.172**	0.200	0.213	0.171	**0.163**

Table 3 (continued)

Dataset	Uniform				Sequential feature selection			
Index	GA	PSO	ALO	LALO	SFFS	SFBS	SFS	SBS
1	0.378	0.470	0.372	**0.369**	0.390	0.396	0.391	0.391
2	0.272	0.272	**0.268**	0.272	0.768	0.768	0.824	0.824
3	0.209	0.243	**0.156**	**0.156**	0.138	0.188	0.142	0.232
4	0.066	0.042	**0.024**	**0.024**	0.046	0.036	0.076	0.062
5	0.108	0.121	**0.042**	**0.042**	0.044	0.090	0.080	0.099
6	**0.147**	**0.147**	0.148	**0.147**	0.199	0.236	0.282	0.219
7	0.284	0.372	0.255	**0.249**	0.347	0.347	0.320	0.336
8	0.326	**0.326**	0.339	**0.326**	0.535	0.428	0.513	0.514
9	0.050	0.143	**0.047**	0.050	0.174	0.176	0.176	0.291
10	0.192	0.148	0.094	**0.080**	0.090	0.042	0.131	0.093
11	**0.538**	**0.538**	0.539	**0.538**	0.811	0.811	0.613	0.613
12	0.044	**0.044**	0.045	**0.044**	0.053	0.055	0.059	0.065
13	0.096	0.069	**0.059**	**0.059**	0.143	0.110	0.169	0.131
14	0.257	0.253	0.237	**0.234**	0.296	0.267	0.303	0.315
15	0.360	0.332	0.311	**0.305**	0.379	0.377	0.328	0.347
16	0.052	0.065	0.018	**0.015**	0.146	0.173	0.129	0.186
17	0.447	**0.447**	0.448	**0.447**	0.494	0.461	0.483	0.496
18	0.186	0.190	0.094	**0.089**	0.608	0.382	0.510	0.475
19	0.145	0.144	0.143	**0.140**	0.661	0.615	0.701	0.637
20	0.047	0.049	0.047	**0.046**	0.075	0.075	0.078	0.075
21	0.203	0.216	0.177	**0.170**	0.280	0.281	0.279	0.289

Bold values represent the best (smallest) results obtained for each dataset

T test is statistical importance measure that demonstrates regardless of whether the contrast between two groups' midpoints in all probability mirrors a *real* distinction in the population from which the groups were inspected [52].

The above four measures are applied directly to the fitness function obtained based on the validation set.

To assess the future performance of whole feature selection model, the following two indicators are used:

Mean test error is used to evaluate the performance of the feature selection on the data that the classifier never sees.

Average Fisher score is a measure that assesses the feature subset like the distances between data points in various classes are as big as possible, while the distances between data points in the same class are as small as possible [53]. Fisher score in this work is calculated for individual features given the class labels as in Eq. (24).

$$F_j = \frac{\sum_{k=1}^{c} n_k(\mu_k^j - \mu^j)^2}{(\sigma^j)^2}, \tag{24}$$

where F_j is the Fisher index for feature j, μ^j, $(\sigma^j)^2$ is the mean and standard deviation of the whole dataset, n_k is the size of class k, and μ_k^j is the average of class k.

The average F-score is calculated as the average of score sum of optimal solution found by running individual optimizers for M times.

Average run time is used in this study to assess the average processing time for the different methods used.

5.3 Results and discussion

The methods adopted in this work are deterministic single-solution methods, namely sequential forward selection (SFS), sequential backward selection (SBS) and their floating version SFFS and SFBS based on the version proposed in [54]. Also, the two common multi-agent optimizers are adopted namely genetic algorithms (GA) [55] and particle swarm optimization (PSO) [56].

Table 3 outlines the mean fitness acquired over the different runs of the various multi-agent optimizers as well as the same fitness calculated for selected features by the

Small	Large
$\begin{bmatrix} 0&1&0&0&0&0&0&1&0\\ 1&0&0&0&0&0&0&1&0\\ 0&0&1&0&0&0&0&0&1\\ 0&0&0&0&0&1&1&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&1&0&0&1&0&1 \end{bmatrix}$	$\begin{bmatrix} 0&1&1&1&0&1&1&1&1\\ 1&0&1&1&1&1&1&0&1\\ 0&1&1&0&1&1&1&1&1\\ 1&0&1&1&1&0&1&1&1\\ 1&1&1&1&1&1&0&1&1\\ 1&1&0&1&1&1&1&1&1 \end{bmatrix}$

Uniform	Mixed
$\begin{bmatrix} 1&0&1&0&1&1&1&0&0\\ 1&1&0&0&0&0&0&1&1\\ 0&0&1&0&0&0&1&1&1\\ 1&1&1&1&1&1&1&0&0\\ 0&0&0&0&1&0&0&1&1\\ 0&1&1&1&0&1&0&0&0 \end{bmatrix}$	$\begin{bmatrix} 1&0&0&0&1&1&1&0&1\\ 0&0&1&0&1&0&1&1&1\\ 0&1&0&0&1&1&1&1&1\\ 0&0&0&1&1&1&1&1&1\\ 0&0&0&0&1&1&0&1&1\\ 0&0&0&0&0&1&1&1&1 \end{bmatrix}$

MRMR	
$\begin{bmatrix} 0.8&0&0&1&0&0&1\\ 0.1&1&0&1&0&1&1\\ 0&1&1&0&0&1&0\\ 0.1&0&1&1&1&0&0\\ 0&1&1&0&1&1&0\\ 0&0&0&0&1&0&1 \end{bmatrix}$	

Fig. 6 Sample initial wolves positions using different initialization with 9 search agents and 6 dimensions

Table 5 Standard deviation of fitness calculated for multi-agent optimizers over various datasets using uniform initialization

Dataset	Uniform			
index	GA	PSO	ALO	LALO
1	0.013	0.020	0.012	**0.010**
2	0.067	0.067	**0.063**	0.067
3	0.047	0.044	0.039	**0.024**
4	0.016	0.017	**0.009**	**0.009**
5	0.016	0.016	**0.014**	0.016
6	0.025	**0.024**	0.026	**0.024**
7	0.091	0.091	**0.081**	0.091
8	0.057	0.054	0.057	**0.051**
9	0.020	0.020	**0.010**	0.020
10	0.056	0.055	0.049	**0.039**
11	0.158	0.158	**0.153**	0.158
12	0.004	**0.004**	0.005	**0.004**
13	0.010	0.012	0.009	**0.008**
14	0.025	0.019	0.019	**0.017**
15	0.037	0.044	0.036	**0.036**
16	0.018	0.022	0.017	**0.013**
17	0.021	**0.020**	0.022	**0.020**
18	0.087	0.093	0.086	**0.069**
19	0.031	0.029	0.028	**0.021**
20	0.003	0.004	0.004	0.004
21	0.093	0.093	**0.082**	0.084

Bold values represent the best (smallest) results obtained for each dataset

Table 4 The *p* value for the *t* test for the fitness measures calculated for stochastic-multi-agent optimizers versus the proposed LALO

Init	GA	PSO	ALO
Uniform	0.043	0.045	0.045
Small	0.039	0.040	0.041
Large	0.049	0.049	0.050
Mixed	0.009	0.012	0.013
MRMR	0.05	0.05	0.061

deterministic single-solution methods. We can remark from the table that the optimal solution obtained from the proposed LALO is much better than the optimal solution of the other optimization methods on the same number of iterations. We can also remark that regardless of the used initializer, the LALO can reach much better optima than the ALO. The enhanced performance can be interpreted by the fact that the LALO generates a set of random solutions that are apart from the current optimal solution in each iteration allowing the optimizer to escape from local optima solution. Such diversity of solutions even at the last iterations of optimization can be interpreted by the thick tail of the Lèvy distribution. Besides, the LALO can generate diverse locations for the search agents even if it is initialized with nearby solutions as in the case of large initialization method, while the ALO can reach such goal.́ One also can remark the impact of the initialization on the performance of all multi-agent optimizers. On the other hand, we can observe that the forward and backward iterative methods can easily be stuck in the first found optimal solution in the search space. Also, the nesting problem found in such methods which make it difficult to abandon a feature that is selected as in SFS and SBS.

The MRMR initializer places a search agent very close to the global optimal solution as it initializes such agent using entropy and correlation constraints. Therefore, we can remark the enhance in the performance of the MRMR method over the other methods. We can also remark that although the MRMR initializes a search agent with information related to the data; mutual information, it is still not optimal as it is not directly related to the classifier performance

Table 6 Standard deviation of fitness calculated for multi-agent optimizers over various datasets

Dataset	Small				Large			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.021	0.013	**0.011**	**0.011**	0.019	0.016	**0.010**	0.011
2	0.078	**0.078**	**0.078**	**0.078**	0.072	0.072	**0.059**	0.072
3	0.026	0.026	**0.019**	0.026	0.040	0.045	0.038	**0.036**
4	0.013	0.013	**0.009**	0.010	0.017	0.013	**0.009**	0.010
5	0.021	0.021	**0.013**	0.021	0.022	0.018	**0.014**	0.016
6	**0.022**	**0.022**	0.025	**0.022**	0.025	0.025	**0.024**	0.025
7	0.063	0.070	0.059	**0.057**	0.085	0.085	**0.049**	0.085
8	0.052	0.052	**0.051**	0.052	0.055	0.055	**0.048**	0.055
9	0.016	0.017	**0.010**	0.016	0.022	0.022	0.015	**0.011**
10	0.045	0.045	**0.039**	0.045	0.038	0.042	**0.033**	0.037
11	**0.163**	**0.163**	0.165	**0.163**	0.160	0.160	**0.148**	0.160
12	**0.004**	**0.004**	0.005	**0.004**	**0.004**	**0.004**	0.005	**0.004**
13	0.010	0.019	**0.008**	0.009	0.017	0.010	0.009	**0.008**
14	0.029	0.029	0.021	**0.017**	0.020	**0.018**	**0.018**	**0.018**
15	0.053	0.051	0.044	**0.039**	0.039	0.040	**0.037**	0.038
16	0.019	0.019	0.015	**0.014**	0.024	0.021	0.017	**0.015**
17	0.024	0.024	**0.020**	0.024	**0.020**	**0.020**	**0.020**	**0.020**
18	0.072	0.076	**0.069**	0.072	0.085	0.090	0.079	**0.061**
19	0.014	0.037	0.018	0.040	0.029	0.029	0.032	**0.023**
20	0.005	**0.001**	0.002	0.006	**0.005**	**0.005**	**0.005**	0.006
21	0.115	0.104	**0.067**	0.096	0.080	**0.079**	0.089	0.096
Dataset	Mixed				MRMR			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.018	0.019	**0.011**	**0.011**	0.012	0.016	0.012	**0.010**
2	0.090	0.097	0.099	**0.060**	0.086	0.080	0.090	**0.066**
3	0.038	0.047	**0.036**	0.037	0.028	0.035	0.026	**0.022**
4	0.018	0.015	**0.009**	0.010	0.016	0.019	**0.009**	**0.009**
5	0.022	0.022	**0.014**	0.016	0.016	0.018	**0.012**	0.014
6	**0.025**	0.026	0.026	**0.025**	**0.024**	**0.024**	**0.024**	**0.024**
7	0.076	0.076	**0.036**	0.076	0.081	0.081	**0.049**	0.081
8	**0.055**	**0.055**	**0.055**	**0.055**	**0.054**	**0.054**	**0.054**	**0.054**
9	0.020	0.013	0.012	**0.011**	0.011	0.010	**0.009**	**0.009**
10	0.038	0.042	**0.036**	**0.036**	0.047	0.047	0.037	**0.028**
11	0.163	**0.162**	0.164	**0.162**	0.165	0.160	0.168	**0.159**
12	**0.004**	**0.004**	**0.004**	**0.004**	**0.004**	**0.004**	**0.004**	**0.004**
13	0.016	0.014	**0.009**	**0.009**	0.011	0.021	0.009	**0.008**
14	0.027	0.025	0.020	**0.018**	0.024	0.023	0.020	**0.019**
15	0.051	0.047	0.045	**0.038**	0.041	0.048	0.040	**0.039**
16	0.018	0.017	0.014	**0.013**	0.018	0.019	**0.012**	0.014
17	**0.021**	**0.021**	**0.021**	**0.021**	**0.020**	**0.020**	0.022	**0.020**
18	0.075	0.078	**0.067**	0.075	0.069	0.069	**0.062**	0.069
19	0.060	0.018	0.039	**0.000**	0.024	0.023	**0.017**	0.024
20	0.011	**0.001**	**0.001**	**0.001**	0.017	**0.001**	0.029	0.003
21	0.107	0.090	0.089	**0.075**	0.091	0.085	0.098	**0.075**

Bold values represent the best (smallest) results obtained for each dataset

Table 7 Average feature size using different initializers for all multi-agent optimizers over various datasets and feature size ratio for the deterministic single-solution methods

Dataset	Small				Large			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.774	0.753	0.729	**0.596**	0.740	0.761	0.721	**0.700**
2	0.447	0.447	**0.424**	0.447	0.543	0.543	**0.489**	0.543
3	0.276	0.243	**0.193**	0.243	0.371	0.371	**0.214**	0.371
4	0.690	0.706	0.626	**0.474**	0.733	0.720	0.630	**0.593**
5	0.329	0.345	**0.223**	0.329	0.490	0.490	**0.260**	0.490
6	**0.676**	**0.676**	0.714	**0.676**	**0.700**	**0.700**	**0.700**	**0.700**
7	0.484	0.481	0.428	**0.300**	0.642	0.751	0.631	**0.628**
8	0.487	**0.470**	0.496	**0.470**	**0.504**	**0.504**	0.511	**0.504**
9	0.721	0.765	0.642	**0.486**	0.705	0.690	0.684	**0.638**
10	0.861	0.839	0.751	**0.603**	0.830	0.780	0.731	**0.664**
11	0.178	0.178	**0.161**	0.178	0.242	0.242	**0.169**	0.242
12	**0.487**	**0.487**	0.500	**0.487**	0.530	0.530	**0.513**	0.530
13	0.554	0.560	0.516	**0.453**	0.575	0.543	**0.514**	0.542
14	0.817	0.805	0.719	**0.556**	0.848	0.798	0.763	**0.678**
15	0.730	0.712	0.680	**0.573**	0.734	0.787	0.700	**0.627**
16	0.566	0.559	0.521	**0.464**	0.556	0.598	0.536	**0.528**
17	0.826	**0.767**	0.850	**0.767**	0.821	0.821	0.821	0.821
18	0.506	0.561	0.492	**0.400**	0.486	0.494	0.463	**0.452**
19	0.479	0.456	0.488	**0.422**	0.825	0.813	0.392	**0.353**
20	0.479	0.446	0.464	**0.408**	0.801	0.801	0.390	**0.378**
21	0.432	0.416	0.170	**0.137**	0.673	0.650	0.131	**0.122**
Dataset	Mixed				MRMR			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.734	0.762	**0.696**	0.729	0.808	0.822	0.758	**0.633**
2	0.510	0.510	**0.434**	0.510	0.485	0.485	**0.452**	0.485
3	0.352	0.352	**0.186**	0.352	0.300	0.309	**0.183**	0.300
4	0.665	0.598	0.585	**0.570**	0.679	0.628	0.585	**0.519**
5	0.390	0.390	**0.219**	0.390	0.340	0.344	**0.235**	0.340
6	**0.671**	0.698	0.705	**0.671**	**0.652**	**0.652**	0.671	**0.652**
7	0.611	0.632	**0.567**	0.569	0.592	0.701	0.577	**0.523**
8	**0.485**	**0.485**	0.500	**0.485**	**0.470**	0.474	0.478	**0.470**
9	0.773	0.811	0.681	**0.581**	0.507	0.538	**0.492**	0.496
10	0.756	0.827	0.721	**0.633**	0.580	0.601	0.515	**0.495**
11	0.189	0.189	**0.161**	0.189	0.253	0.253	**0.186**	0.253
12	**0.533**	**0.533**	0.550	**0.533**	**0.487**	0.503	0.533	**0.487**
13	0.542	0.514	**0.482**	0.514	0.539	0.576	0.484	**0.449**
14	0.740	0.755	0.704	**0.648**	0.773	0.800	0.719	**0.611**
15	0.758	0.811	0.690	**0.633**	0.666	0.647	0.570	**0.560**
16	0.576	0.655	0.536	**0.526**	0.590	0.637	0.551	**0.487**
17	**0.808**	**0.808**	0.846	**0.808**	0.784	0.827	0.833	0.775
18	0.525	0.559	0.479	**0.448**	0.612	0.551	0.506	**0.427**
19	0.467	0.476	0.473	**0.419**	0.477	0.488	0.482	**0.409**
20	0.476	0.466	0.472	**0.411**	0.472	0.476	0.466	**0.423**
21	0.437	0.424	0.178	**0.126**	0.446	0.418	0.173	**0.131**

Table 7 (continued)

Dataset	Uniform				Sequential feature selection			
index	GA	PSO	ALO	LALO	SFFS	SFBS	SFS	SBS
1	0.826	0.832	0.738	**0.617**	0.542	0.667	0.500	0.542
2	0.464	0.464	**0.462**	0.464	0.016	0.016	0.016	0.016
3	0.298	0.298	**0.193**	0.298	0.071	0.262	0.071	0.262
4	0.655	0.646	0.596	**0.544**	0.333	0.296	0.222	0.481
5	0.344	0.344	**0.235**	0.344	0.083	0.146	0.063	0.104
6	**0.662**	0.740	0.743	**0.662**	0.333	0.333	0.286	0.333
7	0.722	0.731	0.613	**0.521**	0.462	0.487	0.538	0.589
8	0.482	**0.474**	0.489	**0.474**	0.185	0.185	0.148	0.148
9	0.697	0.691	0.594	**0.507**	0.324	0.148	0.287	0.102
10	0.676	0.780	0.674	**0.587**	0.590	0.513	0.667	0.564
11	0.228	0.228	**0.164**	0.228	0.083	0.083	0.083	0.083
12	0.491	0.497	0.577	**0.487**	0.533	0.467	0.367	0.333
13	0.568	0.584	0.509	**0.444**	0.211	0.211	0.211	0.211
14	0.816	0.788	0.719	**0.607**	0.593	0.852	0.704	0.593
15	0.776	0.713	0.677	**0.580**	0.300	0.400	0.333	0.367
16	0.633	0.596	0.564	**0.479**	0.179	0.205	0.154	0.231
17	0.856	**0.813**	0.871	**0.813**	0.500	0.583	0.5	0.458
18	0.484	0.490	0.460	**0.408**	0.063	0.167	0.063	0.188
19	0.480	0.480	0.484	**0.420**	0.003	0.006	0.003	0.006
20	0.471	0.460	0.459	**0.413**	0.028	0.028	0.038	0.028
21	0.446	0.413	0.171	**0.128**	0.030	0.219	0.219	0.229

Bold values represent the best (smallest) results obtained for each dataset

used in wrapper-based feature selection. Moreover, as can be seen, we may remark those initializers with a diversity of population; uniform initializer and mixed initializer can perform better than initializers with less diversity, small and large initializers. For ensuring the significance in advance of the performance, the *p* value for the *t* test is calculated for the two optimizers and outlined in Table 4. We can remark that there is a significant enhancement in the performance of the LALO over the ALO at a significance level of 5% using small, large, mixed and uniform initializers. Smaller significance can be remarked for using the MRMR initializer which can be interpreted by the fact that the MRMR always initializes the optimizers with a solution close to the optimal solution and hence it eases the optimization task for the optimizers.

The standard deviation of optimal fitness function value obtained over all the runs of the optimizer can be used as an indicator for optimizer *repeatability* of results and performance *robustness*. Such measure is depicted in Tables 5 and 6 for all optimizers over all the datasets. We can notice from the table that both antlion optimizers have comparable standard deviation which proves that even if we use a random distribution with infinite variance, we still can get repeatable results due to the used selection capability in the LALO. Furthermore, we can see that the LALO and ALO has a general standard deviation less than the ones obtained for the GA and PSO which proves the capability of ALO and LALO to converge to optimal or near-optimal solution regardless of the used random exploration and exploitation used and due to the adaptive control of the step size/exploration rate adopted in both methods.

The secondary goal in the used fitness function is the selected feature size. This indicator is calculated and depicted in Table 7. We can remark that this objective is also minimized and the LALO is still keeping its superiority in performance in comparison to ALO, PSO, and GA. The addition of such secondary objective complicates the shape of the fitness function which is successfully minimized by the proposed LALO but allows the optimizer to *carefully* add extra features to maximize the classification performance. We can recognize that deterministic single-solution methods have in general better performance in this indicator which is natural for the very cautious move of such algorithms but of course on the price of classification accuracy.

Table 8 depicts the performance of the optimizers on the test data that the classifier never sees. We can remark

Table 8 Mean test error using different initializers for all multi-agent optimizers over various datasets and test error for deterministic single-solution methods

Dataset	Small				Large			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.419	0.428	**0.384**	0.385	0.460	0.415	**0.383**	**0.383**
2	0.431	**0.391**	0.446	**0.391**	**0.373**	**0.373**	0.390	**0.373**
3	0.216	0.232	**0.157**	0.161	0.258	0.271	**0.172**	0.185
4	0.066	0.133	**0.040**	0.047	0.140	0.093	**0.041**	0.042
5	0.069	0.171	**0.059**	0.069	0.112	0.168	**0.061**	0.074
6	0.186	0.186	**0.185**	0.186	**0.178**	**0.178**	**0.178**	**0.178**
7	0.339	0.338	0.315	**0.303**	0.369	0.402	0.310	**0.272**
8	**0.392**	**0.392**	0.396	**0.392**	**0.396**	**0.396**	0.405	**0.396**
9	0.128	0.143	**0.059**	0.060	0.079	0.156	**0.059**	**0.059**
10	0.153	0.128	0.117	**0.109**	0.158	0.208	0.117	**0.105**
11	0.610	0.610	**0.608**	0.610	0.604	0.604	**0.599**	0.604
12	**0.047**	**0.047**	**0.047**	**0.047**	**0.046**	**0.046**	0.047	**0.046**
13	0.134	0.169	0.066	**0.064**	0.144	0.160	**0.065**	0.067
14	0.319	0.307	**0.249**	0.258	0.295	0.377	**0.263**	0.265
15	0.392	0.395	0.348	**0.343**	0.409	0.395	**0.356**	**0.356**
16	0.073	0.148	**0.067**	0.073	0.091	0.163	0.082	**0.066**
17	0.468	0.468	**0.463**	0.468	**0.462**	**0.462**	0.465	**0.462**
18	0.202	0.222	0.160	**0.140**	0.178	0.264	0.164	**0.130**
19	0.228	0.206	**0.204**	**0.204**	0.232	**0.231**	**0.231**	**0.231**
20	0.055	0.047	0.066	**0.040**	0.049	0.049	0.051	**0.048**
21	0.310	0.344	0.309	**0.287**	0.400	0.400	0.404	**0.361**
Dataset	Mixed				MRMR			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.433	0.461	**0.380**	0.382	0.396	0.476	**0.384**	**0.384**
2	**0.388**	**0.388**	0.418	**0.388**	0.374	**0.360**	0.395	**0.360**
3	0.254	0.197	**0.174**	0.184	0.179	0.275	0.164	**0.159**
4	0.067	0.103	**0.044**	**0.044**	0.064	0.115	0.045	**0.044**
5	0.110	0.078	**0.060**	0.069	0.064	0.063	**0.060**	0.063
6	**0.181**	**0.181**	**0.181**	**0.181**	**0.179**	**0.179**	**0.179**	**0.179**
7	0.375	0.420	0.318	**0.277**	0.350	0.426	0.322	**0.282**
8	**0.399**	**0.399**	0.412	**0.399**	**0.393**	**0.393**	0.403	**0.393**
9	0.136	0.174	0.062	**0.053**	0.121	0.152	0.051	**0.049**
10	0.173	0.198	0.114	**0.089**	0.089	0.064	0.039	**0.038**
11	**0.607**	**0.607**	0.614	**0.607**	**0.603**	**0.603**	0.609	**0.603**
12	**0.047**	**0.047**	**0.047**	**0.047**	**0.047**	**0.047**	**0.047**	**0.047**
13	0.076	0.173	0.068	**0.066**	0.078	0.070	**0.066**	**0.066**
14	0.328	0.384	0.265	**0.259**	0.319	0.344	0.270	**0.261**
15	0.425	0.359	0.359	**0.355**	0.429	0.387	**0.344**	0.346
16	0.139	0.170	**0.066**	**0.066**	0.165	0.084	0.078	**0.069**
17	**0.461**	**0.461**	**0.461**	**0.461**	0.469	0.469	**0.464**	0.469
18	0.248	0.269	0.150	**0.148**	0.153	0.155	**0.132**	0.145
19	0.225	0.213	0.222	**0.204**	0.224	0.213	**0.206**	0.208
20	0.047	**0.038**	0.047	0.046	**0.043**	0.047	0.045	0.044
21	0.323	0.362	0.303	**0.301**	0.301	0.343	0.313	**0.290**

Table 8 (continued)

Dataset	Uniform				Sequential feature selection			
index	GA	PSO	ALO	LALO	SFFS	SFBS	SFS	SBS
1	0.414	0.410	**0.382**	**0.382**	0.381	0.392	0.386	0.391
2	0.361	0.361	**0.357**	0.361	0.784	0.784	0.830	0.830
3	0.260	0.239	**0.166**	0.169	0.143	0.217	0.145	0.219
4	0.110	0.099	0.045	**0.043**	0.053	0.049	0.0587	0.047
5	0.111	0.118	**0.063**	**0.063**	0.044	0.093	0.080	0.110
6	0.178	0.178	**0.175**	0.178	0.237	0.252	0.251	0.235
7	0.339	0.377	0.285	**0.271**	0.370	0.352	0.331	0.342
8	**0.402**	**0.402**	0.411	**0.402**	0.571	0.457	0.566	0.551
9	0.110	0.135	**0.056**	**0.056**	0.177	0.178	0.166	0.293
10	0.178	0.156	0.106	**0.097**	0.107	0.048	0.125	0.079
11	**0.594**	**0.594**	0.603	**0.594**	0.703	0.703	0.667	0.667
12	**0.047**	**0.047**	0.048	**0.047**	0.062	0.065	0.0543	0.056
13	0.088	0.097	0.066	**0.065**	0.142	0.120	0.169	0.126
14	0.272	0.335	**0.261**	0.265	0.285	0.279	0.282	0.290
15	0.361	0.420	**0.350**	0.351	0.420	0.376	0.375	0.371
16	0.087	0.145	**0.080**	0.087	0.152	0.170	0.124	0.162
17	**0.463**	**0.463**	0.464	**0.463**	0.478	0.467	0.499	0.505
18	0.222	0.209	0.177	**0.173**	0.355	0.391	0.563	0.564
19	0.227	0.216	0.209	**0.208**	0.671	0.606	0.714	0.657
20	0.049	0.048	0.049	**0.047**	0.075	0.075	0.076	0.075
21	0.308	0.346	0.318	**0.298**	0.274	0.245	0.246	0.246

Bold values represent the best (smallest) results obtained for each dataset

that the selected features from the LALO are much fit to be a candidate for future performance thanks to the design of the fitness function and the capability of LALO to optimize such function. Such performance proves the ability of the proposed LALO to adaptively search the space of feature for an optimal combination of features maximizing classification performance.

Another indicator for assessing the future performance of the selected features is depicted in Table 9. The Fisher index calculates the data compactness versus feature span which eases the classifier design based on the selected features. We can observe that the feature set selected by the LALO is much better thanks to the capability of LALO to search the design fitness function for the global optima.

A final indicator that is related to the runtime is depicted in Table 10 where the runtime for each optimizer is measured and sum over all the used initializers. We can remark from the table that the runtime for LALO is much better than the ALO thanks to the simple implementation of the Lèvy random number generator rather than the complete random walk provided in the implementation of ALO published in [32]. Moreover, in contrast to sequentially adding or removing features as in SFS, SBS, SFFS, SFBS, and multi-agent optimization methods can add the batch of features. Hence, LALO can provide enhanced time performance while keeping better classification performance as we saw before.

Table 9 Average F-score using different initializers for all multi-agent optimizers over various datasets and Fisher calculated for the deterministic single-solution methods

Dataset	Small				Large			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	0.399	**0.423**	0.381	0.310	**0.409**	0.395	0.379	0.362
2	4.013	3.995	**4.049**	3.783	**4.619**	**4.619**	3.942	**4.619**
3	0.094	**0.095**	0.088	0.091	0.132	**0.188**	0.081	0.104
4	0.881	0.850	0.829	**0.920**	0.915	0.848	0.839	**0.969**
5	0.196	0.242	0.162	**0.284**	**0.238**	**0.238**	0.172	**0.238**
6	1.258	1.236	1.276	**1.174**	1.256	1.256	1.243	**1.257**
7	0.022	0.109	0.001	**0.111**	0.010	0.055	0.002	**0.062**
8	0.296	0.296	0.248	0.310	0.283	0.288	**0.290**	0.283
9	0.090	0.141	0.022	**0.023**	0.114	0.131	0.023	**0.024**
10	0.078	0.096	0.031	**0.039**	0.084	0.055	**0.085**	**0.085**
11	**1129.500**	**1129.500**	979.600	**1129.500**	**1502.800**	**1502.800**	1084.200	**1502.800**
12	0.114	0.114	0.113	**0.115**	**0.118**	**0.118**	0.116	**0.118**
13	**2.789**	2.749	2.714	2.359	2.409	2.430	**2.375**	2.335
14	**0.080**	0.056	0.006	0.005	**0.035**	0.018	0.006	0.006
15	0.425	0.425	0.369	**0.449**	0.456	0.500	0.381	**0.474**
16	0.593	**0.667**	0.575	0.537	0.566	**0.582**	0.542	0.566
17	0.363	0.363	**0.400**	0.363	**0.394**	**0.394**	0.382	**0.394**
18	**13.000**	12.982	12.970	11.520	**14.020**	**14.020**	11.730	**14.020**
19	0.008	**0.009**	0.008	**0.009**	0.006	0.006	0.007	**0.015**
20	**0.008**	0.007	**0.008**	0.006	0.005	0.006	0.006	**0.012**
21	0.101	0.311	0.128	**0.339**	0.193	0.093	0.103	**0.499**
Dataset	Mixed				MRMR			
index	GA	PSO	ALO	LALO	GA	PSO	ALO	LALO
1	**0.452**	0.435	0.363	0.383	0.420	**0.486**	0.397	0.331
2	**4.670**	**4.670**	3.313	**4.670**	**4.115**	**4.115**	3.251	**4.115**
3	0.152	**0.167**	0.079	0.098	0.130	**0.170**	0.082	0.097
4	0.801	0.881	0.796	**0.834**	0.819	0.820	0.804	**0.889**
5	0.213	0.252	0.158	**0.253**	0.246	0.272	0.168	**0.296**
6	1.203	1.177	1.212	**1.373**	1.213	1.213	1.223	**1.348**
7	0.081	0.084	0.001	**0.091**	0.030	0.045	0.001	**0.091**
8	**0.275**	**0.275**	0.262	**0.275**	**0.305**	**0.305**	0.293	**0.305**
9	0.090	0.070	0.022	**0.091**	0.075	0.104	0.022	**0.092**
10	0.044	0.038	0.041	**0.045**	0.040	0.103	0.031	**0.041**
11	1196.300	1196.300	**993**	1196.300	**1566.900**	**1566.900**	**1160.100**	**1566.900**
12	**0.123**	**0.123**	0.121	**0.123**	0.115	0.112	0.125	**0.132**
13	2.532	2.522	2.371	**2.562**	2.321	**2.335**	2.310	2.198
14	0.070	**0.095**	0.006	0.005	**0.084**	0.016	0.006	0.005
15	**0.415**	0.395	0.382	**0.395**	0.405	0.469	**0.476**	0.383
16	**0.625**	0.575	0.560	0.554	**0.662**	0.627	0.602	0.531
17	0.382	0.382	**0.399**	0.382	0.377	0.377	**0.389**	0.377
18	13.668	13.662	**13.670**	12.850	14.443	**14.466**	14.400	12.210
19	0.009	0.008	0.010	**0.017**	0.009	0.009	0.009	**0.010**
20	0.008	**0.009**	0.006	0.006	**0.008**	**0.008**	0.007	**0.008**
21	0.100	0.310	0.129	**0.339**	0.101	0.313	0.129	**0.340**

Table 9 (continued)

Dataset	Uniform				Sequential feature selection			
index	GA	PSO	ALO	LALO	SFFS	SFBS	SFS	SBS
1	0.480	0.450	0.382	0.324	0.508	0.527	0.543	0.534
2	4.387	4.387	3.365	**4.487**	0.408	0.408	0.345	0.345
3	0.162	0.133	0.083	0.097	1.079	0.597	1.078	0.395
4	0.854	0.828	0.803	**0.866**	1.632	1.566	1.632	1.200
5	0.193	0.276	**0.362**	0.193	1.441	1.011	1.433	1.245
6	1.282	1.312	1.322	**1.495**	2.284	2.364	2.776	1.971
7	0.015	0.015	0.002	**0.021**	0.002	0.003	0.002	0.002
8	0.288	0.288	**0.289**	**0.289**	0.479	0.739	0.852	0.965
9	0.116	0.042	0.022	0.021	0.050	0.079	0.050	0.063
10	0.061	**0.106**	0.031	0.030	0.052	0.060	0.047	0.054
11	1482.200	**1482.2**	1007.300	**1482.2**	567.524	567.524	567.524	567.524
12	0.112	0.125	0.131	**0.132**	0.271	0.259	0.331	0.247
13	2.484	**2.549**	2.437	2.143	6.431	5.536	6.335	8.723
14	0.071	**0.093**	0.006	0.005	0.010	0.008	0.009	0.009
15	0.385	0.471	0.382	**0.388**	1.107	0.848	0.959	0.881
16	0.655	0.579	0.571	**0.657**	1.610	1.141	1.536	1.396
17	0.394	0.394	0.413	**0.494**	0.512	0.474	0.540	0.488
18	12.835	**12.862**	12.750	11.240	1.205	2.147	1.771	12.398
19	0.009	0.009	0.008	**0.019**	0.853	0.872	0.922	0.703
20	**0.007**	**0.007**	0.006	0.006	0.032	0.032	0.033	0.032
21	0.100	0.313	0.128	**0.340**	0.046	0.019	0.018	0.019

Bold values represent the best (biggest) results obtained for each dataset

Table 10 Average runtime using uniform initialization for all multi-agent optimizers over various datasets and for the deterministic single-solution methods

Index	SFS	SBS	SFFS	SFBS	GA	PSO	ALO	LALO
1	13.991	12.824	**8.696**	18.626	41.066	41.051	41.039	40.226
2	65.920	65.920	25.397	45.690	10.394	10.165	10.457	**7.996**
3	54.573	68.838	184.865	89.929	6.410	6.613	**6.388**	6.613
4	8.385	15.059	**6.185**	14.733	7.486	7.527	7.327	7.014
5	66.849	80.176	220.777	115.131	5.933	6.105	**5.916**	6.105
6	54.916	69.384	869.112	996.519	**6.506**	6.543	6.673	6.543
7	55.248	57.553	60.149	153.747	9.524	9.560	9.452	**9.444**
8	159.835	180.284	2967.129	349.799	6.747	6.677	6.755	**6.677**
9	142.751	275.323	1368.826	1139.919	70.005	70.187	69.944	**61.740**
10	26.648	25.192	12.926	983.206	10.577	10.450	10.445	**10.259**
11	0.340	0.340	**0.470**	10.690	6.557	6.896	6.700	6.896
12	7.049	10.594	137.638	**15.371**	27.898	27.703	27.996	27.169
13	65.179	103.511	547.105	1179.365	25.941	26.013	25.818	**24.001**
14	26.591	26.269	28.437	561.369	10.067	9.975	9.906	**9.020**
15	173.876	280.320	3564.951	8716.965	11.622	11.733	11.612	**11.225**
16	167.725	205.083	1013.593	1283.789	5.671	5.653	5.617	**5.314**
17	34.625	34.218	763.158	1771.110	23.730	23.658	23.738	**23.083**
18	95.083	32.916	991.538	61.309	5.955	5.913	5.991	**5.533**
19	1488.630	21.670	962.982	19.679	23.294	24.098	**23.142**	24.023
20	3997.502	3987.502	957.502	3987.502	47.911	48.321	47.699	**46.923**
21	32747.066	34747.066	2757.066	32747.066	6.968	6.711	**6.331**	6.976

Bold values represent the best (smallest) results obtained for each dataset

6 Conclusions and future work

A novel variant of ALO is proposed based on Lèvy flights. This study applies the Lèvy antlion optimization (LALO) algorithm for feature selection and is tested on 21 different datasets, compared its result to the original ALO. Five initialization methods are used to assess the local and global searching capability of each algorithm. The results are evaluated using different indicators assessing convergence, repeatability value, and feature reduction size, performance on test data and statistical significance as well as runtime.

We can conclude the following:

- Lèvy flights help the ALO to search the space efficiently for the optimal solution avoiding premature convergence and stagnation.
- The proposed LALO can provide repeatable results with efficient performance even at complex fitness function shapes.
- The initial positions of search agents affect the performance of the different optimizers and generally, initializers that provide initial diverse population are better. Moreover, initializers such as MRMR that initialize the search agents with solutions that have many desirable properties has better performance.

Acknowledgments This work was partially supported by the IPROCOM Marie Curie initial training network, funded through the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement No. 316555.

References

1. Guyon I, Elisseeff A (2003) An introduction to variable and attribute selection. Mach Learn Res 3:1157–1182
2. Dash M, Liu H (1997) Feature selection for classification. Intell Data Anal J 1(3):131–156
3. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif. Intell. 97(1):273–324
4. Chuang LY, Tsai SW, Yang CH (2011) Improved binary particle swarm optimization using catsh effect for attribute selection. Expert Syst Appl 38:12699–12707
5. Xue B, Zhang M, Browne WN (2014) Particle swarm optimisation for feature selection in classification: novel initialisation and updating mechanisms. Appl Soft Comput 18:261–276
6. Whitney A (1971) A direct method of nonparametric measurement selection. IEEE Trans Comput C–20(9):1100–1103
7. Marill T, Green D (1963) On the effectiveness of receptors in recognition systems. IEEE Trans Inf Theory IT–9(1):11–17
8. Xue B, Zhang M, Browne WN (2013) Particle swarm optimization for feature selection in classification: a multi-objective approach. IEEE Trans Cybern 43(6):1656–1671
9. Daolio F, Liefooghe A, Verel S, Aguirre H, Tanaka K (2015) Global vs local search on multi-objective NK-landscapes: contrasting the impact of problem features. In: Conference on genetic and evolutionary computation (GECCO), pp 559–566
10. Valdez F (2015) Bio-inspired optimization methods. In: Springer handbook of, computational intelligence, pp 1533–1538
11. Emary E, Zawbaa HM (2016) Impact of chaos functions on modern swarm optimizers. PLoS ONE 11(7):e0158738
12. Xue B, Zhang M, Browne WN, Yao X (2016) A survey on evolutionary computation approaches to feature selection. IEEE Trans Evolut Comput 20(4):606–626
13. Mirjalili S, Saremi S, Mirjalili SM, Coelho LDS (2016) Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. Expert Syst Appl 47:106–119
14. Emary E, Zawbaa HM, Grosan C (2017) Experienced Grey Wolf optimizer through reinforcement learning and neural networks. IEEE Trans Neural Netw Learn Syst (TNNLS) 99:1–14
15. Mittal N, Singh U, Sohi BS (2016) Modified Grey Wolf optimizer for global engineering optimization. Appl Comput Intell Soft Comput 2016:8
16. Shoghian S, Kouzehgar M (2012) A comparison among Wolf Pack search and four other optimization algorithms. World Academy of Science, Engineering and Technology, vol 6
17. Segura C, Rionda SB, Aguirre AH, Pena SIV (2015) A novel diversity-based evolutionary algorithm for the traveling salesman problem. In: Conference on genetic and evolutionary computation (GECCO), pp 489–496
18. Goncalves EC, Plastino A, Freitas AA (2015) Simpler is better: a novel genetic algorithm to induce compact multi-label chain classifiers. In: Conference on genetic and evolutionary computation (GECCO), pp 559–566
19. Holland J (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
20. Chakraborty B (2002) Genetic algorithm with fuzzy fitness function for feature selection. In: IEEE international symposium on industrial electronics vol 1, pp 315–319
21. Zhu Z, Ong YS, Dash M (2007) Wrapper-filter feature selection algorithm using a memetic framework. IEEE Trans Syst Man Cybern 37(1):70–76
22. Eiben AE, Raue PE, Ruttkay Z (1994) Genetic algorithms with multi-parent recombination. In: Conference on evolutionary computation, third conference on parallel problem solving from nature, vol 866, pp 78–87
23. Wang X, Yang J, Teng X, Xia W, Jensen R (2007) Feature selection based on rough sets and particle swarm optimization. Pattern Recognit Lett 28:459–471
24. Ming H (2008) A rough set based hybrid method to feature selection. In: International symposium on knowledge acquisition and modeling, pp 585–588
25. Akbari R, Mohammadi A, Ziarati K (2010) A novel bee swarm optimization algorithm for numerical function optimization. Commun Nonlinear Sci Numer Simul 15(10):3142–3155
26. Maeda M, Tsuda S (2015) Reduction of artificial bee colony algorithm for global optimization. Neurocomputing 148:70–74
27. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Glob. Optim. 39(3):459–471
28. Yang XS (2005) Engineering optimizations via nature-inspired virtual bee algorithms. In: Artificial intelligence and knowledge engineering applications, first international work-conference on the interplay between natural and artificial computation, pp 317–323
29. Sundareswaran K, Sreedevi VT (2008) Development of novel optimization procedure based on honey bee foraging behavior. In: IEEE international conference on systems, man and cybernetics, pp 1220–1225

30. Li XL, Shao ZJ, Qian JX (2002) An optimizing method based on autonomous animates: Fish-swarm algorithm. Methods Pract Syst Eng 22:32–38
31. Emary E, Zawbaa HM, Hassanien AE (2016) Binary Grey Wolf optimization approaches for feature selection. Neurocomputing, Elsevier 172:371–381
32. Mirjalili S (2015) The ant lion optimizer. Adv. Eng. Softw. 83:83–98
33. Meiri R, Zahavi J (2006) Using simulated annealing to optimize the feature selection problem in marketing applications. Eur J Oper Res 171(3):842–858
34. Pavlyukevich I (2007) Levy flights, non-local search and simulated annealing. Computat Phys 226:1830–1844
35. Reynolds AM, Frye MA (2007) Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search. PLoS ONE 2(4):e354
36. Viswanathan GM (2008) Lèvy flights and superdiffusion in the context of biological encounters and random searches. Phys Life Rev 5:133–150
37. Barthelemy P, Bertolotti J, Wiersma DS (2008) A Levy flight for light. Nature 453:495–498
38. Zawbaa HM, Emary E, PARV B (2015) Feature selection based on antlion optimization algorithm. In: 3rd world conference on complex systems (WCCS), Morocco, pp 1–7
39. Emary E, Zawbaa HM, Hassanien AE (2016) Binary ant lion approaches for feature selection. Neurocomputing, Elsevier 213:54–65
40. Zawbaa HM, Emary E, Grosan C (2016) Feature selection via chaotic antlion optimization. PLoS ONE 11(3):e0150652
41. Yang XS (2010) Nature-inspired, metaheuristic algorithms. Luniver Press, Cambridge
42. Yang XS, Cui Z, Xiao R, Gandomi AH, Karamanoglu M (2013) swarm intelligence and bio-inspired computation. Elsevier, London, pp 18–20
43. Bhandari AK, Singh VK, Kumar A, Singh GK (2014) Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy. Expert Syst Appl 41:3538–3560
44. Tang J, Zhao X (2009) Particle swarm optimization with adaptive mutation. In: International conference on information engineering, vol 2, pp 234–237
45. Vieira SM, Sousa JMC, Runkler TA (2010) Two cooperative ant colonies for feature selection using fuzzy models. Expert Syst Appl 37:2714–2723
46. Frank A, Asuncion A (2010) UCI machine learning repository
47. Hastie T, Tibshirani R, Friedman J (2001) The Elements of Statistical Learning. Springer, Berlin
48. Akadi AE, Amine A, Ouardighi AE, Aboutajdine D (2011) A two-stage gene selection scheme utilizing MRMR filter and GA wrapper. Knowl Inf Syst 26(3):487–500
49. Chuang LY, Chang HW, Tu CJ, Yang CH (2008) Improved binary PSO for feature selection using gene expression data. Comput Biol Chem 32(1):29–38
50. Tilahun SL, Ong HC (2015) Prey-Predator algorithm: a new metaheuristic algorithm for optimization problems. Inf Technol Decis Mak 14(6):1331–1352
51. Jia L, Gong W, Wu H (2009) An improved self-adaptive control parameter of differential evolution for global optimization. Comput Intell Intell Syst 51:215–224
52. Rice JA (2006) Mathematical Statistics and Data Analysis, 3rd edn. Duxbury Advanced
53. Duda RO, Hart PE, Stork DG (2000) Pattern classification, 2nd edn. Wiley-Interscience Publication, London
54. Kotropoulos VC (2008) Fast and accurate feature subset selection applied into speech emotion recognition. Signal Process 88(12):2956–2970
55. Yang CH, Tu CJ, Chang JY, Liu HH (2006) Po–Chang Ko, dimensionality reduction using GA-PSO. In: Joint conference on information sciences (JCIS), Atlantis Press, Taiwan
56. Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Sixth international symposium on micro machine and human science, Japan, pp 39–43