Neural Comput & Applic (2018) 30:2355–2369
https://doi.org/10.1007/s00521-016-2818-2

ORIGINAL ARTICLE

A multi-verse optimizer approach for feature selection and optimizing SVM parameters based on a robust system architecture

Hossam Faris[1] · Mohammad A. Hassonah[1] · Ala' M. Al-Zoubi[1] · Seyedali Mirjalili[2] · Ibrahim Aljarah[1]

Received: 21 September 2016 / Accepted: 19 December 2016 / Published online: 2 January 2017

Abstract Support vector machine (SVM) is a well-regarded machine learning algorithm widely applied to classification tasks and regression problems. SVM was founded based on the statistical learning theory and structural risk minimization. Despite the high prediction rate of this technique in a wide range of real applications, the efficiency of SVM and its classification accuracy highly depends on the parameter setting as well as the subset feature selection. This work proposes a robust approach based on a recent nature-inspired metaheuristic called multi-verse optimizer (MVO) for selecting optimal features and optimizing the parameters of SVM simultaneously. In fact, the MVO algorithm is employed as a tuner to manipulate the main parameters of SVM and find the optimal set of features for this classifier. The proposed approach is implemented and tested on two different system architectures. MVO is benchmarked and compared with four classic and recent metaheuristic algorithms using ten binary and multi-class labeled datasets. Experimental results demonstrate that MVO can effectively reduce the number of features while maintaining a high prediction accuracy.

Keywords Optimization · SVM · Support vector machines · Multi-verse optimizer · MVO · Feature selection · Metaheuristics

✉ Hossam Faris
hossam.faris@ju.edu.jo; 7ossam@gmail.com

Mohammad A. Hassonah
mohammad.a.hassonah@gmail.com

Ala' M. Al-Zoubi
alaah14@gmail.com

Seyedali Mirjalili
seyedali.mirjalili@griffithuni.edu.au

Ibrahim Aljarah
i.aljarah@ju.edu.jo

1 Business Information Technology Department, King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan

2 School of Information and Communication Technology, Griffith University, Nathan, Brisbane, QLD 4111, Australia

1 Introduction

Support vector machine (SVM) is a supervised machine learning model designed for analyzing the data and recognizing certain visible or hidden patterns. SVM can be used for either classification or regression analysis. SVM was first designed and proposed by Vladimir Vapnik [39]. Besides linear classification, SVM can efficiently perform nonlinear classification by projecting the training dataset into a higher dimensional space so the categories of the training data are separated by a determined hyperplane. In the literature, SVM showed high prediction accuracy and modeling capability in wide range of real classification, pattern recognition and regression problems [22, 27, 30–32, 34–36].

Different kernel functions were used and applied by researchers in the literature (e.g., linear, polynomial, or sigmoid). Radial Basis Function (RBF) kernel (also known as Gaussian function) is the most popular and highly recommended kernel. As reported in several studies, RBF kernel helps SVM to achieve an accurate prediction and reliable performance [4, 15, 49]. In addition, RBF can lead a better analysis of higher dimensional data and it has less parameters to optimize [13].

In order to get the advantage of SVM and achieve the best generalization ability with maximum prediction power, two important problems should be addressed. The first problem is the optimization of the error penalty parameter C of SVM and its kernel parameters. The second one is the selection of the best representative subset of features that will be used in the training process. Regarding the former problem, conventionally, SVM parameters are selected using an exhaustive grid search algorithm. However, this method suffers from a long running time due to the need for a huge number of possible evaluations [37, 50]. Therefore, there are researchers who proposed other efficient solutions for optimizing the SVM and the parameters of its kernel. A common type of these approaches is the metaheuristic algorithms. In the literature, Metaheuristic algorithms showed high efficiency in generating acceptable solutions when the problem is very complex and the search space is extremely large [40–44]. Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) are very popular examples of the metaheuristic search algorithms. GA was designed by John Holland and inspired by the Darwinian theories of evolution and natural selection [11, 12, 21]. On the other hand, PSO is a swarm intelligent-based algorithm inspired by the movement of bird and fish flocks in nature [17–19]. Some examples that deployed GA, PSO and other nature-inspired metaheuristic algorithms in optimizing SVM can be found in [1, 4, 14, 29, 47, 52].

Feature selection is a process of choosing a set of M features from a data set of N features, $M < N$, so that the value of some evaluation function or criterion is optimized over the space of all possible feature subsets. The goal of the feature selection process is to eliminate the irrelevant features and consequently decreasing the training time and reducing the complexity of the developed classification models [5, 9, 25]. It was reported in several studies that feature selection occasionally leads to improvements in the predictive accuracy and enhancements in the comprehensibility and generalization of the developed model [2, 26]. On the other side, selecting the best subset of features from all possible 2^N subsets is not trivial and turns to be NP-hard problem when the search space grows [2]. SVM is not different from other data mining and machine learning techniques and its performance could be highly improved by applying the feature selection process.

Previous works proposed different techniques for optimizing SVM parameters and performing feature selection simultaneously. One of the first attempts was made by Huang and Wang [15], in which GA was applied to this problem. Based on comparisons with the conventional grid search algorithm using different datasets, they showed that GA is able to optimize SVM to reach better accuracy with a fewer number of features. A similar approach was followed by Lin et al. [24] where the PSO algorithm was employed instead of GA. They compared their results with those obtained by Huang and Wang [15]. Their results showed that PSO was very competitive compared to GA outperforming it in six datasets out of ten. Another work was conducted by Zhao et al. [51] and they used GA with feature chromosome operation for subset feature selection and optimizing SVM parameters. In their work, almost a similar system architecture was experimented and compared to those in [15, 24].

In this paper, we propose, experiment and discuss a robust approach based on the recent multi-verse optimizer (MVO) for feature selection and optimizing the parameters of SVM in order to maximize the accuracy of SVM. According to our knowledge, this is the first time to employ MVO for optimizing SVM. This work also considers the proposal of an improved architecture to improve the generalization power and robustness of the SVM. The proposed model is named as MVO+SVM. MVO is a metaheuristic search algorithm inspired by a number of cosmological theories including the Big Bang theory [28]. MVO has shown high competency and efficiency when applied to challenging optimization problems such as training feed-forward neural networks [7]. The proposed MVO-based SVM is evaluated based on seven binary datasets and three multi-class datasets selected from the UCI machine learning repository. The MVO+SVM approach is designed and applied using two different system architectures. The first one is identical to those employed in [15, 24], while the second architecture is proposed in order to maximize the generalization ability of the model and therefore achieving more robust results. In addition, evaluation results of MVO-based SVM are compared with those obtained for GA, PSO and two recent metaheuristic algorithms which are the Firefly Algorithm (FF) [45] and Bat Algorithm (BAT) [46]. In the past few years, BAT and FF were investigated for optimizing SVM in many studies [4, 6, 16, 48]. All our experiments are carried out using the two aforementioned system architectures.

This paper is structured as follows: Sect. 2 briefly describes the SVM algorithm. Section 3 presents the structure of the MVO algorithm. The proposed approach for feature selection and optimizing the parameters of SVM is provided and discussed in Sect. 4. The conducted experiments and the results are discussed and analyzed in Sect. 5. At last, the conclusion and future works are summarized in Sect. 6.

2 Support vector machines

SVM is a mathematical model and a powerful universal approximator proposed and developed by Vapnik [38]. SVM can be used for both classification and regression tasks. Recently, in an extensive comparative study [8], it was shown that SVM stands among the best classifiers implemented so far. Rather than the minimizing the empirical error like in Neural Networks, the theoretical foundations of SVM are derived from the structural risk minimization idea [39].

As in the majority of classifiers, SVM depends on the training process to build its model. By using the kernel trick, SVM transforms the training data by nonlinear mapping functions to a higher dimensional space where the data can be separated linearly, or to find the best hyperplanes (support vectors) with maximal normalized margin with respect to the data points. Therefore, the goal of the learning process of SVM is to look for the optimal linear hyperplanes in that dimension [10]. Figure 1 depicts an example of a binary class dataset separated by SVM optimal hyperplanes.

Suppose we have a dataset $\{x_i, y_j\}_{i=1,\ldots,n}$ where the $x_i \in \Re^{d}$ represents the input features, d is number of features in the training dataset and $y_i \in \Re$ is its corresponding actual output, the main target of the SVM algorithm is to draw the linear decision function given in Eq. (1):

$$f(x) = \langle w, \phi_i(x) \rangle + b \tag{1}$$

w and b a weight and a constant, respectively, which have to be estimated from the dataset. ϕ is a nonlinear function which maps the input features to higher feature space. $\langle .,. \rangle$ indicates the dot product in $\Re^{d}$.

This problem can be represented to minimize the following function:

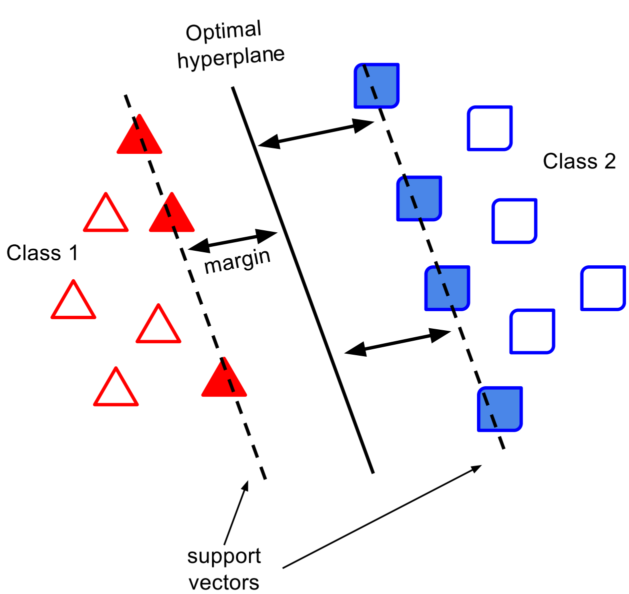

Fig. 1 Optimal hyperplane in support vector machine

$$R(C) = \frac{C}{n}\sum_{i=1}^{n} L_{\varepsilon}(f(x_i), y_i) + \frac{1}{2}\|w\|^2$$

$L_{\varepsilon}(f(x_i), y_i)$ is known as $\varepsilon-$intensive loss function which can be represented as shown in Eq. (2):

$$L_{\varepsilon}(f(x), y) = \begin{cases} |f(x) - y| - \varepsilon & |f(x) - y| \geq \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

By adding the slack variables ξ_i and ξ_i^*, the problem can be formulated to minimize Eq. (3) in subject to the constraints given in Eq. (4).

$$R(w, \xi_i^*) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{3}$$

$$\begin{cases} y_i - \langle w, x_i \rangle - b & \leq \varepsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i & \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq 0 \end{cases} \tag{4}$$

C represents a penalty for a prediction error that is greater than ε. ξ_i and ξ_i^* are slack variables that measure the error cost based on the training data.

This optimization problem with the specified constraints can be handled by means of Lagrangian multipliers as a quadratic optimization problem. The solution can be represented as given in Eq. (5).

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(x_i, x) + b \tag{5}$$

where α_i and α_i^* are Lagrange multipliers which are subject to the following constraints:

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$$

$$0 \leq \alpha_i \leq C \quad i = 1, \ldots, n$$

$$0 \leq \alpha_i^* \leq C \quad i = 1, \ldots, n$$

$K(.)$ is the kernel function and its value is an inner product of two vectors x_i and x_j in the feature space $\phi(x_i)$ and $\phi(x_j)$. $K(.)$ can be represented as shown in Eq. (6).

$$K(x_i, x_j) = \phi(x_i).\phi(x_j) \tag{6}$$

The most popular and used kernel functions in the literature are the Polynomial, Hyperbolic Tangent Kernel and the RBF Kernel as given in Eqs. (7), (8) and (9), respectively.

$$K_p(x_i, x_j) = \langle x_i, x_j + 1 \rangle^d \tag{7}$$

$$K_h(x_i, x_j) = \tan h(c_1(x_i.x_j) + c_2) \tag{8}$$

$$K_{rbf}(x_i, x_j) = \exp(-\gamma||x_j - x_i||^2), \text{ where } \gamma > 0 \tag{9}$$

One of the key issues here is that selection of kernel functions and the values of their parameters have a great impact on the accuracy of the SVM model. The values of

SVM parameters have high influence of its performance. As we discussed in Sect. 1, there are different approaches for optimizing these parameters. Since the metaheuristic algorithms are very efficient to find optimal values for optimization problems, we employ MVO to do that for the first time in the literature.

3 Multi-verse optimizer

The multi-verse optimizer [28] is a recent evolutionary metaheuristic algorithm, which mimics the rules in one of the theories of multi-verse. The main inspiration of this algorithm comes from the theory of existence of multiple universes and their interactions via black, white, and worm holes. This algorithm is a population-based stochastic algorithm and approximates the global optimum for optimization problems with a collection of solutions.

In this algorithm, two parameters should be calculated first to update the solutions: Wormhole Existence Probability (WEP) and Traveling Distance Rate (TDR). Such parameters dictate how often and how much the solutions change during the optimization process and defined as follows:

$$WEP = a + t \times \left(\frac{b-a}{T}\right) \tag{10}$$

where a is the minimum, b is the maximum, t is the current iteration, and T represents the maximum number of iterations allowed.

$$TDR = 1 - \frac{t^{1/p}}{T^{1/p}} \tag{11}$$

where p defines the exploitation accuracy.

The main parameter of TDR is p. The exploitation is emphasized proportional to the value of this parameter.

After calculating WEP and TDR, the position of solutions can be updated using the following equation:

$$x_i^j = \begin{cases} \begin{cases} x_j + TDR + ((ub_j - lb_j) * r_4 + lb_j)) & \text{if} \quad r_3 < 0.5 \\ x_j - TDR + ((ub_j - lb_j) * r_4 + lb_j)) & \text{if} \quad r_3 \geq 0.5 \end{cases} & \text{if} \quad r_2 < WEP \\ x^j_{RouletteWheel} & \text{if} \quad r_2 \geq WEP \end{cases} \tag{12}$$

where X_j is the jth element of the best individual, WEP, TDR are coefficients, lb_i and ub_i are the lower and upper bounds of the jth element, r_2, r_3, r_4 are randomly generated numbers drawn from the interval of [0, 1], x_i^j represents the jth parameter in ith individual, and $x^j_{\text{RouletteWheel}}$ is the jth element of a solution picked by the roulette wheel selection mechanism.

This equation shows that the position of solution can be updated with respect to the current best individual obtained using the WEP. If the r_3, which is a random number in [0, 1], less than 0.5, the solution is required to get the value of the jth dimension in the best solution. WEP is increased during optimization, so this is how the MVO increases the exploitation of the best solution obtained so far.

The exploration and local optima avoidance are guaranteed with the second part of the above equation, in which the jth variable in the solution i is replaced with that in a selected solution using a roulette wheel. When using the second part, the current solution is considered to have a black hole, and the one of the best solutions contain a white hole.

The white holes are chosen with a roulette wheel proportional to their fitness value. The black holes are created inversely proportional to the fitness value for minimization problems. This mechanism assists the MVO algorithm to improve the poor solutions using the best solutions over the course of iterations. Since the solutions exchange variables, there are sudden changes in the solutions and consequently improved exploration. If a solution stagnates in a local optimum, this approach is able to revolve it as well.

To balance between exploration and exploitation, WEP and TDR should be changed adaptively using Eqs. (10) and (11). In this work we have used $a = 0.2$, $b = 1$, and $p = 6$ in these equations. Figure 2 shows how the WEP and TDR change over the course of iterations.

The MVO algorithm first generates a set of random solutions and calculates their corresponding objectives. The position of solutions is repeatedly updated using Eq. (12) until the satisfaction of an end condition. Meanwhile, the random parameter (r_2, r_3, r_4), WEP, and TDR are updated for each solution. It has been proved that this algorithm is able to provide very comparative and occasionally superior results compared to the current approaches. In this following section, this algorithm is integrated to the SVM for the first time.

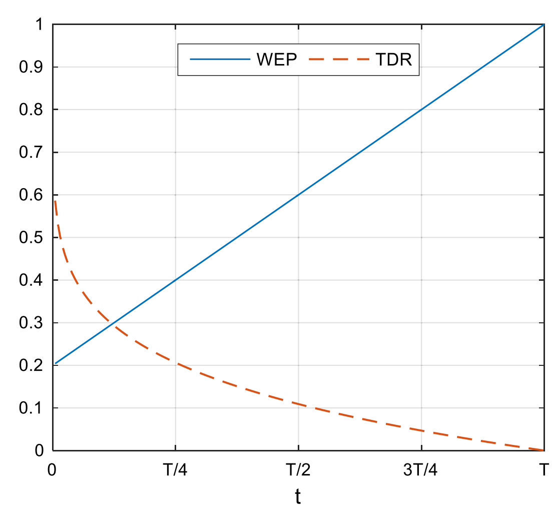

Fig. 2 WEP and TDR

4 Proposed MVO–SVM model

In this section, we describe three important points regarding the proposed implementation of MVO for feature selection and SVM parameters optimization. They are the encoding scheme used to represent MVO universes, the fitness function, and the system architectures followed in this work. The key points are described as follows:

4.1 Encoding scheme

In the two architectures investigated in this work, the individuals are encoded as a vector of real numbers. The number of elements in each vector equals number of features in the dataset plus two elements to represent SVM parameters: the Cost (C) and Gamma (γ). The implemented encoding scheme is shown in Fig. 3. Each element in the vector is a randomly generated number in the interval [0,1]. Therefor, the elements that represent features are rounded: if the element is larger or equal to 0.5, its value is rounded to 1 and the feature is selected, otherwise the value is rounded to 0 and the feature is not selected. For the C and γ, those parameters need to be mapped to different scales since their search space is different. For example, the value of the element corresponding to C is mapped to the interval [0, 35000] while the element corresponding to γ is mapped to [0, 32]. In our implementation, the values of C and γ are linearly transformed using Eq. (13).

$$B = \frac{A - min_A}{max_A - min_A}(max_B - min_B) + min_B. \tag{13}$$

4.2 Fitness evaluation

In order to assess the generated universes (solutions), we rely on the confusion matrix shown in Fig. 4 which is considered as the primary source for evaluating classification models. Based on this confusion matrix, the classification accuracy rate is calculated as given in Eq. (14):

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}. \tag{14}$$

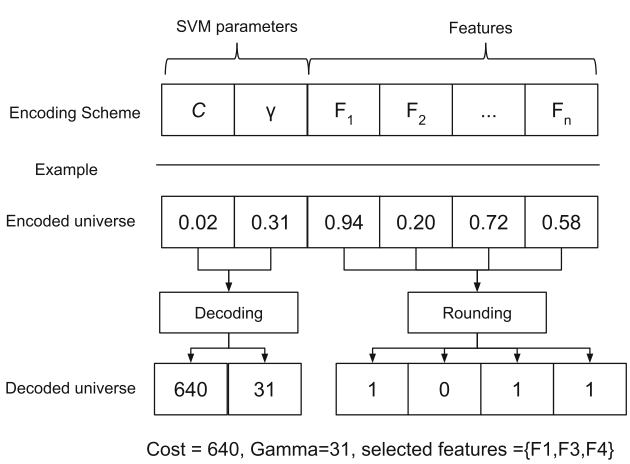

Fig. 3 Encoding scheme of individuals for SVM optimization and feature selection

		Actual class	
		Positive	Negative
Predicted class	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Fig. 4 Confusion matrix

4.3 System architectures

This subsection describes the main system architectures that are applied to perform feature selection and optimizing the parameters of SVM simultaneously using the MVO algorithm. The term "system architecture" was used in the previous studies to describe the processes that are carried out to perform this task and their sequence. In this work two different architectures are utilized. The first system architecture used and implemented in [15], while the second is a modified version of the first one. We will refer to the two architectures as "Architecture I" and "Architecture II," respectively. In other words, we propose Architecture II as an approach to enhance the generalization ability and robustness of the developed model. To describe the workflow of these architectures, we provide the following bullet points and figures:

- Data normalization: this is a preprocessing step performed on the features of all datasets. The values of all features are mapped into same scale in order to eliminate the effect of some features that have different range values on the learning process of the algorithm. Therefore, all features are given an equal weight and normalized to fall in the interval [0,1] using a Eq. (15), which is a special form of Eq. (13).

$$B = \frac{A - min_A}{max_A - min_A} \tag{15}$$

- Decoding of universes: the generated vectors (universes) by MVO are split into two parts: the first two elements of the vector correspond to the SVM parameters and they are converted using Eq. (13). The rest of the elements, which correspond to the selected features, are rounded to form a binary vector.

- Select feature subset: after decoding the universe as described earlier to a binary vector, the corresponding features are selected from the training dataset.
- Fitness evaluation: every generated solution by MVO which represents the parameters of SVM and a set of selected features is assessed using the fitness function.
- Termination criterion: the evolutionary cycle of MVO keeps until a termination condition is met. Here we set a maximum number of iterations as a termination condition.
- Reproduction operators: this is a sequence of operators which are applied by MVO in order to evolve the generated universes searching for a better quality solution.

The main differences between those architectures are the fitness evaluation and the methodology for training and testing. In the objective function of Architecture I, the whole training dataset is deployed to build the SVM model, and then the returned fitness of the objective function is the evaluation result of the trained SVM based on the testing part. On the other side, in the objective function of Architecture II, the training part is split again into a number of smaller parts to perform *k*-folds cross-validation. So the SVM is trained *k* times and the average evaluation is returned. In the latter case, the testing part is not presented to the SVM during the iteration of the metaheuristic but used to assess the final selected subset of features and the best obtained parameters. Figures 5 and 6 clearly show the details and differences between the two architectures.

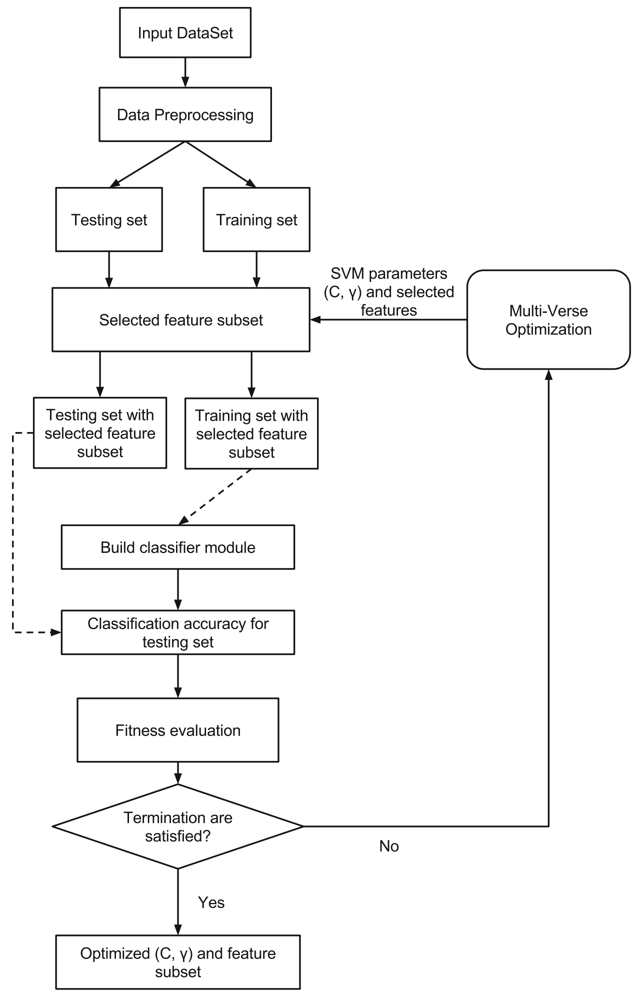

Fig. 5 System architecture I

5 Experiments and results

5.1 Experiments setup

The experiments in this work are conducted on a personal machine with Intel Core i7 processor, 2.40 GHz, 8 GB RAM, using Windows 10 as the operating system. We also used Matlab R2015a (8.5.0.197613) environment as an implementation for our experiment. The LIBSVM implementation is used for the SVM classifier [3].

The proposed MVO–SVM approach is tested and evaluated based on ten datasets drawn from the UCI repository[1] [23]. Seven of the datasets belong to binary class labeled: Heart, Ionosphere, German, Sonar, Breast cancer, Parkinsons and Spectf; while Vowel, Wine and Vehicle are multiclass labeled. Number of features and instances in each dataset is given in Table 1.

The initial parameters of MVO, GA, PSO, BAT and FF algorithms are set as listen in Table 2. The number of universes, individuals and swarm size is identical in all algorithms and set to 30. Generally speaking, there should be special considerations when setting the maximum iterations of the algorithm. This is because that the type of the incorporated feature selection method is a wrapper one. Although wrapper-based feature selection methods are powerful due to the interaction between the selection of the features and the classifier wrapped within the search method, they have the disadvantage of being computationally expensive and they can overfit [20, 33]. Therefore, empirically, it was found that running the algorithm up to a small number of iterations (e.g., 50) can reduce the computation time of the metaheuristic algorithms, and they can converge to a solution.

As mentioned earlier, system Architecture I and II differ in the training/testing methodology implemented in each one. In Architecture I, the cross-validation is set to 10. This means that SVM is trained 10 times where in each time SVM is trained using different 9-folds, and then, the fitness function returns a fitness value based on the 10th testing

1 http://archive.ics.uci.edu/ml/.

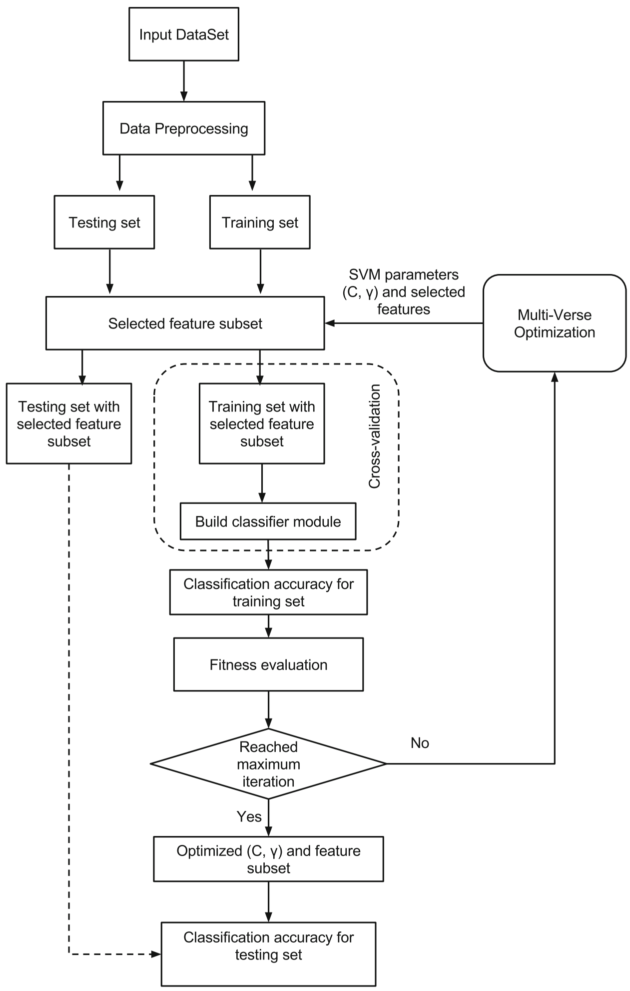

Fig. 6 System architecture II

Table 1 List of datasets

Dataset	Features	Instances	Classes
Heart	13	270	2
Ionosphere	34	351	2
Sonar	60	208	2
German	24	1000	2
Vowel	10	528	11
Wine	13	178	3
Vehicle	18	846	4
Breast Cancer	10	683	2
Parkinsons	22	195	2
Spectf	44	276	2

fold. In Architecture II, however, the outer cross-validation is set to 10-folds and the inner one is set to 3-folds. The experiments are repeated 10 times to get statistically meaningful results.

Table 2 Initial parameters of the MVO, GA, PSO, BAT and FF

Algorithm	Parameter	Value
MVO	Min wormhole existence ratio	0.2
	Max wormhole existence ratio	1
	Universes	30
	Iterations	50
GA	Crossover ratio	0.9
	Mutation ratio	0.1
	Selection mechanism	Roulette wheel
	Population size	30
	Generations	50
PSO	Acceleration constants	[2.1, 2.1]
	Inertia w	[0.9, 0.6]
	Number of particles	30
	Generations	50
BAT	Loudness	0.5
	Pulse rate	0.5
	Frequency minimum	0
	Frequency maximum	1
FF	Alpha	0.2
	Beta	1
	Gamma	1

5.2 Results of optimizing SVM with feature selection

In this part of the experiments, MVO is evaluated and compared to GA, PSO, BAT and FF in performing feature selection and optimizing the parameters of SVM. The five algorithms are evaluated using the ten datasets described earlier. The average of the accuracy rate and the average number of selected features along with the standard deviation are listed in Tables 3 and 4 for Architecture I and Architecture II, respectively. We use $avg \pm std$ deviation to represent these values.

The results of Architecture I show that MVO achieved the highest average accuracy rates compared to GA, PSO, BAT and FF in 8 out 10 datasets exceeding 99% accuracy rate in six of them. Also, MVO shows lower standard deviation values for most of the datasets. It can also be noticed that the three optimizers have very close results regarding the number of selected features. Figure 7 shows the convergence curves of all optimizers based on the averages of accuracy rates for the 10 runs. In this figure MVO clearly shows its higher convergence speed in 8 out the 10 datasets. The best results and parameters obtained based on Architecture I are listed in Table 5. It can be seen that MVO and GA achieved 100% accuracy

Table 3 Results of architecture I

Algorithm	MVO+SVM		GA+SVM		PSO+SVM		BAT+SVM		FF+SVM	
Dataset	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features
Heart	93.7 ± 4.4	7.4 ± 1.8	92.96 ± 5.35	5.3 ± 1.27	92.59 ± 2.87	6.4 ± 1.36	90.74 ± 6.46	7 ± 1.51	90 ± 5.25	6.1 ± 1.58
Ionosphere	99.15 ± 1.3	17.9 ± 2.55	98.58 ± 1.42	16.3 ± 3.87	98.85 ± 1.4	17.3 ± 1.85	97.73 ± 2.11	18.3 ± 3.41	99.15 ± 1.3	16.9 ± 2.51
Sonar	99.02 ± 1.95	27.2 ± 3.64	98.07 ± 2.36	24.8 ± 4.83	96.67 ± 4.29	30.4 ± 1.5	95.7 ± 5.41	28.8 ± 3.12	97.11 ± 3.17	28.7 ± 3.55
German	79.2 ± 4.19	12.5 ± 4.2	81.3 ± 3.13	11.1 ± 5.45	79.8 ± 2.36	12.2 ± 3.16	77.7 ± 3.98	10.5 ± 2.46	77.5 ± 3	12.2 ± 1.47
Vowel	100 ± 0	6.2 ± 0.87	100 ± 0	6.9 ± 1.14	100 ± 0	7.3 ± 1	99.25 ± 0.92	6.3 ± 1	100 ± 0	6.7 ± 0.64
Wine	100 ± 0	7.4 ± 1.69	100 ± 0	6.3 ± 1.73	100 ± 0	6.9 ± 1.22	100 ± 0	7.2 ± 1.25	100 ± 0	6.9 ± 1.97
Vehicle	89 ± 2.15	12 ± 1.55	86.99 ± 4.06	10 ± 1.67	86.98 ± 3.92	11.7 ± 1.55	86.52 ± 4.79	10.3 ± 2	87.48 ± 2.87	10.9 ± 2.51
Breast cancer	99.27 ± 0.98	4.6 ± 1.11	98.97 ± 1.32	5.5 ± 2.73	99.11 ± 0.98	5 ± 2.05	98.53 ± 1.14	5.5 ± 1.57	98.68 ± 1.03	4.7 ± 1.95
Parkinson	99.5 ± 1.43	11.3 ± 2.09	99.47 ± 1.51	10.2 ± 2.12	100 ± 0	12.3 ± 1.42	99.47 ± 1.51	11.9 ± 1.97	100 ± 0	11.7 ± 2.38
Spectf	95.91 ± 4.01	19.9 ± 2.5	95.16 ± 5.24	18.8 ± 5.24	93.62 ± 2.79	21.4 ± 3.34	91.03 ± 2.82	22.6 ± 2.3	95.13 ± 2.28	21.9 ± 4.83

Table 4 Results of architecture II

Algorithm	MVO+SVM		GA+SVM		PSO+SVM		BAT+SVM		FF+SVM	
Dataset	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features	Accuracy	No. of selected features
Heart	83.33 ± 6.47	6.3 ± 2.41	82.96 ± 4.74	8 ± 4.24	81.48 ± 6.42	8.2 ± 1.33	79.62 ± 4.46	5.8 ± 1.54	78.15 ± 8.36	6.3 ± 2.37
Ionosphere	93.16 ± 4.45	18.7 ± 2.97	90.9 ± 4.49	13.6 ± 3.58	91.44 ± 3.63	17.4 ± 1.96	90.61 ± 4.75	17.4 ± 2.69	92.59 ± 3.19	16.4 ± 2.42
Sonar	89.38 ± 8.9	29.5 ± 3.64	82.73 ± 7.67	25.2 ± 7.47	87.5 ± 6.1	30.9 ± 3.33	87 ± 4.84	30.9 ± 4.78	84.17 ± 7.71	29.4 ± 2.87
German	74.4 ± 1.96	15 ± 1.61	76.3 ± 4.29	15.1 ± 1.64	74 ± 5.76	15.2 ± 1.47	73.5 ± 6.77	11 ± 2.76	70 ± 3.85	10.1 ± 2.98
Vowel	99.62 ± 0.75	7.9 ± 0.83	99.05 ± 1.52	8 ± 1.1	98.86 ± 0.93	8.2 ± 0.87	98.47 ± 2.82	7.3 ± 1.27	99.06 ± 1.93	8 ± 0.63
Wine	98.33 ± 2.55	8.3 ± 1.73	94.93 ± 5.25	7.9 ± 1.58	96.66 ± 5.09	7.9 ± 1.04	94.38 ± 5.04	8.6 ± 1.11	96.63 ± 4.46	8.1 ± 1.45
Vehicle	81.8 ± 3.9	12.6 ± 1.36	81.08 ± 1.98	12.6 ± 2.37	81.09 ± 4.79	12.1 ± 1.37	79.41 ± 7.26	10.9 ± 1.51	78.6 ± 5.26	10.9 ± 1.7
Breast cancer	96.48 ± 2.39	6.2 ± 1.4	96.34 ± 2.48	6.1 ± 1.04	96.34 ± 2.37	6.3 ± 0.78	96.49 ± 1.99	4.9 ± 1.14	95.76 ± 1.88	5.2 ± 1.4
Parkinson	95.92 ± 3.59	12.2 ± 1.9	94.89 ± 5.77	10.3 ± 2.3	93.84 ± 4.24	12.2 ± 2.33	94.34 ± 3.63	12.7 ± 1.96	94.84 ± 3.18	10.6 ± 2.53
Spectf	79.05 ± 7.97	20.1 ± 1.5	79.87 ± 9.3	19.1 ± 3.26	78.29 ± 6.58	21 ± 2.8	78.62 ± 7.25	21.6 ± 3.34	78.65 ± 6.46	20.9 ± 4.7

rate in 8 out of 10 datasets while PSO, BAT and FF managed to reach this rate in 7 datasets.

Inspecting the results of Architecture II in Table 4, it may be observed that the obtained accuracy rates by all optimizers are lower than those for Architecture I. This decrease is due to the incorporated training/testing scheme in this architecture. As it was mentioned before, the testing part in this architecture was not represented or seen during the optimization process. Therefore, the results are expected to be lower but more credible. According to the results obtained by Architecture II in Table 4, the MVO algorithm shows higher average accuracy than the other optimizers in all datasets except in the German and Spectf datasets in which it is ranked second after GA. Moreover,

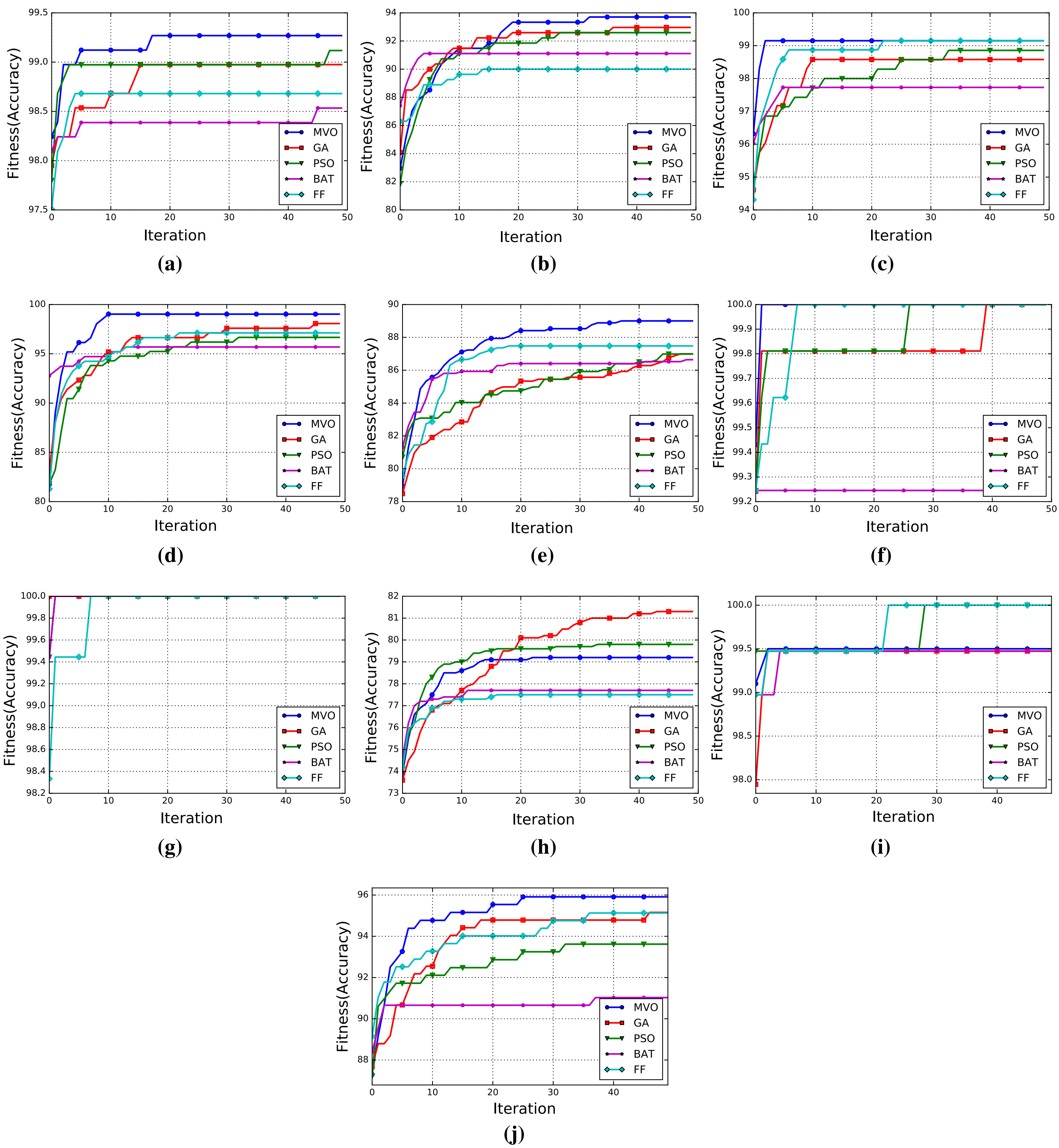

Fig. 7 Convergence curves of MVO, GA, PSO, BAT and FF in optimizing SVM and feature selection based on Architecture I. **a** Breast cancer. **b** Heart. **c** Ionosphere. **d** Sonar. **e** Vehicle. **f** Vowel. **g** Wine. **h** German. **i** Parkinsons. **j** Spectf

Table 5 Best obtained results based on Architecture I

Dataset	MVO+SVM	GA+SVM	PSO+SVM	BAT+SVM	FF+SVM
Heart					
Best accuracy	100	100	96.3	100	96.3
No. of selected features	8	4	4	4	5
Cost (*C*)	31941.4	28267.28	18562.92	16395.42	16265.8
γ	8.43	0.0001	27.01	19.43	12.63
Ionosphere					
Best accuracy	100	100	100	100	100
No. of selected features	17	12	14	15	13
Cost (*C*)	7487.26	1637.64	12386.94	34665.56	22565.59
γ	0.81	1.53	0.87	2.18	0.0001
Sonar					
Best accuracy	100	100	100	100	100
No. of selected features	22	16	30	24	20
Cost (*C*)	10711.62	35000	10102.66	35000	12168.71
γ	1.44	0.9	0.87	0.0001	0.21
German					
Best accuracy	85	89	84	84	83
No. of selected features	15	12	11	15	13
Cost (*C*)	32517.1	10044.12	32700.54	35000	19536.01
γ	0	0.0001	15.76	0.0001	0.0001
Vowel					
Best accuracy	100	100	100	100	100
No. of selected features	5	6	5	5	6
Cost (*C*)	9470.25	8710.15	1322.08	34526.26	10891.24
γ	12.06	13.8	16	23.16	4.17
Wine					
Best accuracy	100	100	100	100	100
No. of selected features	4	4	5	5	4
Cost (*C*)	5149.58	13789.51	32715.96	35000	14755.23
γ	1.48	5.15	3.31	0.0001	7.76
Vehicle					
Best accuracy	91.76	92.86	92.94	94.05	92.86
No. of selected features	11	12	10	12	13
Cost (*C*)	26775.73	19883.93	7429.6	989.31	11950.47
γ	0.39	0.0001	4.17	0.09	0.58
Breast cancer					
Best accuracy	100	100	100	100	100
No. of selected features	3	2	2	4	3
Cost (*C*)	30124.21	9562.28	34005.7	35000	26687.13
γ	24.29	18.56	3.62	0.0001	17.89
Parkinsons					
Best accuracy	100	100	100	100	100
No. of selected features	7	7	10	8	8
Cost (*C*)	4691.48	4071.45	21470.18	2635.98	4275.59
γ	16.17	2.14	10.15	8.24	16.16
Spectf					
Best accuracy	100	100	100	96.3	100
No. of selected features	17	14	18	20	16
Cost (*C*)	2634.96	8045.49	27837.72	3342.66	16232.75
γ	6.66	16.08	28.52	13.82	13.07

MVO shows a higher robustness in most of datasets especially in Ionosphere, Sonar, Vowel and Wine. Figure 8 shows the convergence curves of all optimizers based on Architecture II. In this figure, MVO shows a higher convergence speed in most of the datasets. The best results and parameters obtained based on Architecture II are presented in Table 6. It is shown that only MVO and PSO achieved 100% accuracy rate in 5 out of 10 datasets while BAT, GA and FF comes next with 4, 3, 2 datasets, respectively.

In order to verify the significance of the differences between MVO results and the other optimizers, the non-parametric statistical test Wilcoxon's rank-sum test is

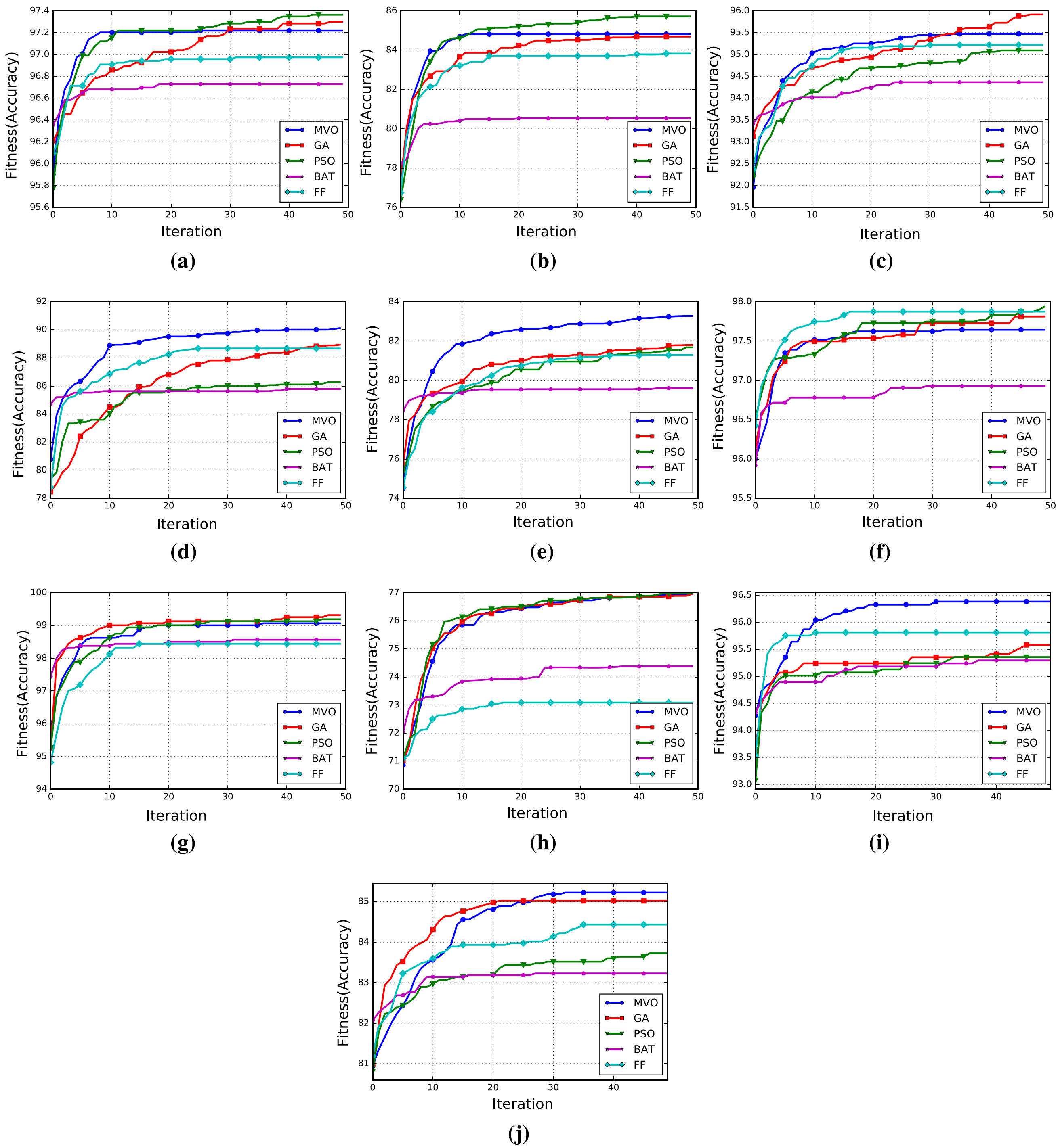

Fig. 8 Convergence curves of MVO, GA, PSO, BAT and FF in optimizing SVM and feature selection based on Architecture II. **a** Breast cancer. **b** Heart. **c** Ionosphere. **d** Sonar. **e** Vehicle. **f** Vowel. **g** Wine. **h** German. **i** Parkinsons. **j** Spectf

Table 6 Best obtained results based on Architecture II

Dataset	MVO+SVM	GA+SVM	PSO+SVM	BAT+SVM	FF+SVM
Heart					
Best accuracy	88.89	88.89	88.89	88.89	92.59
No. of selected features	3	3	7	7	9
Cost (C)	8090.17	16488.46	28811.03	34962.8	28465.52
γ	9.5	0.33	0.0001	0.0002	0.0002
Ionosphere					
Best accuracy	100	97.14	97.14	100	97.14
No. of selected features	19	17	16	15	12
Cost (C)	19636.16	6596.45	33320.6	28035.14	22814.34
γ	1.54	0.64	1.63	1.44	1.92
Sonar					
Best accuracy	100	95.24	100	95.24	95.24
No. of selected features	28	32	34	26	26
Cost (C)	23023.17	35000	29043.67	31331.58	19709.92
γ	0.38	0.25	0.19	0.53	0.64
German					
Best accuracy	78	82	82	84	79
No. of selected features	16	15	15	11	10
Cost (C)	18981.34	6585.16	35000	34948.39	13712.41
γ	0.0001	0.0001	0.0001	0.0007	15.94
Vowel					
Best accuracy	100	100	100	100	100
No. of selected features	7	7	7	7	7
Cost (C)	2567.54	3086.54	35000	854.76	14842.37
γ	2.59	6.16	4.32	2.64	8
Wine					
Best accuracy	100	100	100	100	100
No. of selected features	5	7	7	8	6
Cost (C)	17425.11	11212.74	4106.05	870.14	16346.93
γ	3.57	1.34	1.23	3.09	2.62
Vehicle					
Best accuracy	88.1	83.53	87.06	88.1	87.06
No. of selected features	13	13	11	11	13
Cost (C)	16452.13	29894.9	4262.44	6808.75	12268.22
γ	0.11	0.28	1.77	0.09	0.11
Breast cancer					
Best accuracy	98.55	98.55	100	98.53	98.53
No. of selected features	8	7	6	3	6
Cost (C)	34011.26	925.34	10457.39	34285.81	15715.01
γ	0.0001	0.0001	0.0001	21.96	0.005
Parkinsons					
Best accuracy	100	100	100	100	94.74
No. of selected features	10	7	12	11	10
Cost (C)	7936.23	23834.89	14260.76	15299.07	10308.68
γ	9.11	11.91	5.22	3.09	4.12
Spectf					
Best accuracy	92.59	96.15	88.46	92.59	85.19

Table 6 continued

Dataset	MVO+SVM	GA+SVM	PSO+SVM	BAT+SVM	FF+SVM
No. of selected features	21	27	21	20	14
Cost (*C*)	35000	9511.23	1634.7	6281.27	13492.21
γ	4.65	7.86	15.61	12	27.96

Table 7 *P* values of the Wilcoxon test of MVO classification results versus other algorithms ($p \geq 0.05$ are italized)

	GA	PSO	BAT	FF
Heart	*0.0780*	0.0038	8.66e−10	8.21e−08
Ionosphere	5.03e−05	0.0011	2.29e−05	*0.1671*
Sonar	8.27e−07	0.0192	0.0135	1.60e−05
German	9.39e−06	*0.6211*	*0.5390*	5.55e−21
Vowel	0.0101	6.38e−10	7.94e−04	*0.4848*
Wine	1.38e−08	*0.0575*	1.31e−08	0.0061
Vehicle	0.0013	*0.9031*	0.0199	9.45e−07
Breast cancer	*0.0760*	*0.5367*	*0.7092*	6.52e−06
Parkinsons	*1.00*	4.82e−05	0.0170	*0.2080*
Spectf	*0.2702*	*0.9032*	*1.00*	*0.5390*

carried out. The test is performed based on the results of the MVO against each of the other optimizers at 5% significance level. In Table 7, the *p* values obatined by the test are listed. All p-values values in the table less than 0.05 means that the null hypothesis is rejected (indicating significant difference) at a 5% significance level. According to the results it can be seen that MVO is significantly better than GA and PSO in 5 datasets out of ten. While MVO is significantly better than BAT and FF in 7 and 6 datasets, respectively.

5.3 Comparison with grid search (without feature selection)

In this experiment, we compare MVO with the gird search for optimizing the parameters of SVM. To make the comparison fair, MVO is applied just for parameters optimization without the feature selection part since the grid search does not have this capability. Both techniques were applied with a 10-folds cross- validation. Grid search is used as described in [13, 15]. Table 8 shows the results of comparison based on the aforementioned Arch I and Arch II. The results from Arch I show that MVO is noticeably better than the grid search in 9 datasets. For the Wine dataset, MVO is slightly better while German dataset was the only one that shows better performance for grid search. With checking the average classification accuracies obtained by Arch II, it can been seen that the accuracy rates obtained by MVO are higher than those for the grid search even in the German dataset. The only exception is for Spectf and Breast cancer datasets.

Comparing the results of Arch II to Arch I, two facts can be observed. First, the accuracy rates decreased for MVO and grid search-based SVM models. As mentioned the previous section, this is expected due to the adopted testing strategy. Second, the difference between the accuracy of MVO and the grid search has increased. This can be an evidence that MVO is more robust than the grid search in

Table 8 Comparison between MVO and Grid search in optimizing SVM parameters

	Arch I		Arch II	
	MVO	Grid	MVO	Grid
Heart	87.41 ± 5.02	86.3 ± 4.98	82.96 ± 4.44	80.74 ± 2.22
Ionosphere	96.86 ± 3.25	96.3 ± 1.81	94.03 ± 5.89	92.87 ± 3.21
Sonar	93.69 ± 6.17	89.88 ± 7.25	87.02 ± 5.73	86 ± 5.64
German	75.4 ± 5.99	78.4 ± 3.58	75.8 ± 5.25	75.2 ± 4.45
Vowel	99.81 ± 0.57	91.3 ± 4.33	99.24 ± 0.93	89.97 ± 4.3
Wine	99.44 ± 1.67	99.41 ± 1.76	97.75 ± 3.72	97.19 ± 2.81
Vehicle	88.42 ± 2.74	77.79 ± 3.03	84.99 ± 4.12	75.18 ± 3.25
Breast cancer	98.1 ± 1.96	97.65 ± 1.89	96.63 ± 2.38	96.63 ± 1.61
Parkinsons	97.42 ± 3.49	88.63 ± 5.76	94.89 ± 4.65	86.68 ± 8.71
Spectf	84.72 ± 7.41	79.8 ± 7.19	77.88 ± 5.94	77.91 ± 6.15

optimizing SVM when new and unseen data are presented to the model.

Overall, the results and findings of this section show the merits of the MVO algorithm in improving the performance of SVM. The more accurate results of MVO-based SVM are due to the high exploitation of MVO. In order for SVM to be very accurate, the parameters should be tuned accurately as well. This paper showed that the MVO can be very efficient, which is due the bold role of the best individual obtained using the worm holes in improving the quality of other solutions. The reliability and robustness of the MVO-based SVM originate from the high exploration and local optima avoidance of the MVO algorithm. The sudden changes in the solutions using white/black holes emphasize the exploration process and help in resolving the local optima stagnation. Moreover, the WEP and TDR parameters assist MVO to first explore the search space broadly and, then, exploit the promising regions accurately over the course of iterations. This dynamic control of the exploration and exploitation processes enabled MVO to achieve better results over the other algorithms.

6 Conclusions

In this research work we proposed the application of a recent nature-inspired metaheuristic called multi-verse optimizer for feature selection and optimizing the parameters of SVM simultaneously. Two system architectures were implemented for the proposed approach: the first architecture is commonly used in the literature while the second is proposed in this work to increase the credibility of the SVM prediction results. The developed approach is assessed and benchmarked with four well-regarded metaheuristic algorithms (GA, PSO, BAT and Firefly) and the grid search. Experiments show that MVO was able to optimize SVM achieving the highest accuracy compared with the other optimizers based on the two investigated architectures. The findings and analysis of this work proved the merits of the MVO algorithm in improving the performance of SVM.

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Carrizosa E, Martín-Barragán B, Morales DR (2014) A nested heuristic for parameter tuning in support vector machines. Comput Oper Res 43:328–334
2. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40(1):16–28
3. Chang C-C, Lin C-J (2011) Libsvm: a library for support vector machines. ACM Trans Intell Syst Technol 2(3):27
4. Chao C-F, Horng M-H (2015) The construction of support vector machine classifier using the firefly algorithm. Comput Intell Neurosci 2015:1–8
5. Dash M, Liu H (1997) Feature selection for classification. Intell Data Anal 1(14):131–156
6. Enache A-C, Sgârciu V (2015) Anomaly intrusions detection based on support vector machines with an improved bat algorithm. In: 2015 20th international conference on control systems and computer science. pp 317–321. IEEE
7. Faris H, Aljarah I, Mirjalili S (2016) Training feedforward neural networks using multi-verse optimizer for binary classification problems. Appl Intell 45(2):322–332
8. Fernández-Delgado M, Cernadas E, Barro S, Amorim D (2014) Do we need hundreds of classifiers to solve real world classification problems? J Mach Learn Res 15(1):3133–3181
9. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
10. Han J, Kamber M, Pei J (2011) Data mining: concepts and techniques. Elsevier, Amsterdam
11. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–72
12. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
13. Hsu C-W, Chang C-C, Lin C-J (2003) A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University. http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf. Accessed 1 Aug 2016
14. Zhang H, Wang M, Huang X-H (2010) Parameter selection of support vector regression based on particle swarm optimization. In: 2010 IEEE international conference on granular computing, pp 834–838
15. Huang C-L, Wang C-J (2006) A GA-based feature selection and parameters optimizationfor support vector machines. Expert Syst Appl 31(2):231–240
16. Kavousi-Fard A, Samet H, Marzbani F (2014) A new hybrid modified firefly algorithm and support vector regression model for accurate short term load forecasting. Expert Syst Appl 41(13):6047–6056
17. Kennedy J (1997) The particle swarm: social adaptation of knowledge. In: Proceedings of the 1997 international conference on evolutionary computation, pp 303–308. IEEE Service Center, Piscataway, NJ
18. Kennedy J (1998) The behavior of particles. Evolutionary programming VII, pp 581–587
19. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Neural networks, 1995. Proceedings, IEEE international conference on, vol 4, pp 1942–1948 vol 4. IEEE
20. Ron K, John GH (1997) Wrappers for feature subset selection. Artif Intell 97(1):273–324
21. Krishnakumar K, Goldberg DE (1992) Control system optimization using genetic algorithms. J Guid Control Dyn 15(3):735–740
22. Li X, Lord D, Zhang Y, Xie Y (2008) Predicting motor vehicle crashes using support vector machine models. Accid Anal Prev 40(4):1611–1618
23. Lichman M (2013) UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml. Accessed 1 Aug 2016
24. Lin S-W, Ying K-C, Chen S-C, Lee Z-J (2008) Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Syst Appl 35(4):1817–1824
25. Liu H, Motoda H (1998) Feature extraction, construction and selection: a data mining perspective. Kluwer, Norwell
26. Liu H, Lei Y (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowl Data Eng 17(4):491–502

27. Maglogiannis I, Zafiropoulos E, Anagnostopoulos I (2009) An intelligent system for automated breast cancer diagnosis and prognosis using svm based classifiers. Appl Intell 30(1):24–36
28. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513
29. Qian Q, Gao H, Wang B (2014) A SVM method trained by improved particle swarm optimization for image classification. In: Pattern recognition. Springer, Berlin, pp 263–272
30. Rodan A, Faris H (2015) Echo state network with SVM-readout for customer churn prediction. In: Applied Electrical Engineering and Computing Technologies (AEECT), 2015 IEEE Jordan Conference on, pp 1–5. IEEE
31. Rodan A, Faris H, Alsakran J, Al-Kadi O (2014) A support vector machine approach for churn prediction in telecom industry. Inf-In Interdiscip J 17(8):3961–3970
32. Rodan A, Sheta AF, Faris H (2016) Bidirectional reservoir networks trained using SVM+ privileged information for manufacturing process modeling. Soft Comput. doi:10.1007/s00500-016-2232-9
33. Saeys Y, Inza I, Larrañaga P (2007) A review of feature selection techniques in bioinformatics. Bioinformatics 23(19):2507–2517
34. Shao Y-H, Wang Z, Chen W-J, Deng N-Y (2013) Least squares twin parametric-margin support vector machine for classification. Appl Intell 39(3):451–464
35. Sheta AF, Ahmed SEM, Faris H (2015) A comparison between regression, artificial neural networks and support vector machines for predicting stock market index. Soft Comput 7:8
36. Shin K-S, Lee TS, Kim H-J (2005) An application of support vector machines in bankruptcy prediction model. Expert Syst Appl 28(1):127–135
37. Staelin C (2003) Parameter selection for support vector machines. Hewlett-Packard Company, Tech. Rep. HPL-2002-354R1
38. Vapnik V (1999) An overview of statistical learning theory. IEEE Trans Neural Netw 5:988–999
39. Vapnik V (2013) The nature of statistical learning theory. Springer, Berlin
40. Wang G-G, Deb S, Cui Z (2015) Monarch butterfly optimization. Neural computing and applications, pp 1–20
41. Wang G-G, Gandomi AH, Alavi AH (2014) Stud krill herd algorithm. Neurocomputing 128:363–370
42. Wang G-G, Gandomi AH, Alavi AH, Hao G-S (2014) Hybrid krill herd algorithm with differential evolution for global numerical optimization. Neural Comput Appl 25(2):297–308
43. Wang G-G, Gandomi AH, Zhao X, Chu HCE (2016) Hybridizing harmony search algorithm with cuckoo search for global numerical optimization. Soft Comput 20(1):273–285
44. Wang GG, Deb S, Coelho LDS (2015) Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Int J Bio-Inspired Comput (**in press**)
45. Yang X-S (2010) Firefly algorithm, stochastic test functions and design optimisation. Int J Bio-Inspired Comput 2(2):78–84
46. Yang X-S (2010) A new metaheuristic bat-inspired algorithm. In: *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, Berlin, pp 65–74
47. Yang X-S, Deb S, Fong S (2011) Networked digital technologies: third international conference, NDT 2011, Macau, China, July 11–13, 2011. Proceedings, chapter Accelerated Particle Swarm Optimization and Support Vector Machine for Business Optimization and Applications. Springer, Berlin, pp 53–66
48. Ye Z, Ma L, Wang M, Chen H, Zhao W (2015) Texture image classification based on support vector machine and bat algorithm. In: Intelligent data acquisition and advanced computing systems: technology and applications (IDAACS), 2015 IEEE 8th international conference on, vol 1, pp 309–314. IEEE
49. Xinying Y, Liong S, Babovic V (2004) EC-SVM approach for real-time hydrologic forecasting. J Hydroinform 6:209–223
50. Zhang XL, Chen XF, He ZJ (2010) An ACO-based algorithm for parameter optimization of support vector machines. Expert Syst Appl 37(9):6618–6628
51. Zhao M, Chong F, Ji L, Tang K, Zhou M (2011) Feature selection and parameter optimization for support vector machines: a new approach based on genetic algorithm with feature chromosomes. Expert Syst Appl 38(5):5197–5204
52. Zhou H, Zhao JP, Zheng LG, Wang CL, Cen KF (2012) Modeling No x emissions from coal-fired utility boilers using support vector regression with ant colony optimization. Eng Appl Artif Intell 25(1):147–158