IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 29, 2017, accepted November 28, 2017, date of publication December 4, 2017, date of current version March 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2017.2779794

Optimal Hyper-Parameter Tuning of SVM Classifiers With Application to Medical Diagnosis

ALFONSO ROJAS-DOMÍNGUEZ[1], LUIS CARLOS PADIERNA[1], JUAN MARTÍN CARPIO VALADEZ [1], HECTOR J. PUGA-SOBERANES [1], AND HÉCTOR J. FRAIRE[2]

[1]Tecnológico Nacional de México–Instituto Tecnológico de León, León 37290, México
[2]Tecnológico Nacional de México–Instituto Tecnológico de Ciudad Madero, Ciudad Madero 89460, México

Corresponding author: Luis Carlos Padierna (luiscarlos.padierna@itleon.edu.mx)

The work of A. Rojas-Domínguez was supported by the National Council of Science and Technology of Mexico under Grant CATEDRAS-2598. The work of L. C. Padierna was supported by the National Council of Science and Technology of Mexico under Grant 375524.

ABSTRACT Proper tuning of hyper-parameters is essential to the successful application of SVM-classifiers. Several methods have been used for this problem: grid search, random search, estimation of distribution Algorithms (EDAs), bio-inspired metaheuristics, among others. The objective of this paper is to determine the optimal method among those that recently reported good results: Bat algorithm, Firefly algorithm, Fruit-fly optimization algorithm, particle Swarm optimization, Univariate Marginal Distribution Algorithm (UMDA), and Boltzmann-UMDA. The criteria for optimality include measures of effectiveness, generalization, efficiency, and complexity. Experimental results on 15 medical diagnosis problems reveal that EDAs are the optimal strategy under such criteria. Finally, a novel performance index to guide the optimization process, that improves the generalization of the solutions while maintaining their effectiveness, is presented.

INDEX TERMS Support vector machines, medical diagnosis, heuristic algorithms, particle swarm optimization, density estimation robust algorithm, boltzmann distribution.

I. INTRODUCTION

Support Vector Machines (SVMs) represent a machine learning model used to solve pattern recognition tasks such as classification, regression and outlier detection [1]. The effectiveness of a classifier derived from this model is highly dependent of a kernel function which maps original data to higher dimensional spaces to deal with non-linearly separable data [2]. A kernel function may require a number of parameters to be adjusted by the user; these are commonly known as hyper-parameters of the SVM and their values can affect substantially the performance of SVM classifiers. Properly adjusting the hyper-parameters of a machine learning algorithm requires knowledge of the algorithm, experience, and typically, trial and error. However, in order to systematically and efficiently obtain the best possible solution, this task can be posed as an optimization problem given an adequate objective function that captures the predictive performance of the classifier in terms of the hyper-parameter configurations.

Many different approaches have been shown to be successful in solving the problem of hyper-parameter tuning, from the simplest Grid Search (GS) and Random Search (RS) [3], [4], to more complex metaheuristic algorithms such as evolutionary algorithms [5], bio-inspired algorithms [6]–[9] and estimation of distribution algorithms (EDAs) [10]. Mostly, their success is due to the fact that the number of hyper-parameters is usually small and there are regions (rather than points) of the search space that result in the highest performance of the classifier. Thus, the question to be answered is not whether a particular approach can solve the problem of SVM hyper-parameter tuning, but rather, which of the existing proposed techniques can do it optimally.

To answer this research question, a systematic evaluation of different novel approaches for hyper-parameter tuning is presented, including popular bio-inspired metaheuristics and EDAs because these types of optimization algorithms are the most recently proposed and the most successful. This comparative evaluation is carried out using measures of optimality: effectiveness, efficiency, generalization and complexity.

Usually, the objective function employed to guide the optimization in hyper-parameter tuning of an SVM classifier is the classification accuracy within a cross-validation

scheme. Although this measure provides a good indication of a classifier's performance, it can also be an ambiguous measure of hyper-parameter optimization, because several hyper-parameter configurations can produce classifiers with very similar accuracy. In order to reduce this ambiguity, in this work, a measure of SVMs generalization, the proportion of support vectors (PSV), is employed in conjunction with the classification accuracy to evaluate the effectiveness of the optimization algorithms. Efficiency is measured in terms of the proportion and distribution of function calls (PFC) required by each algorithm; PFC is employed instead of computational time because it is independent of computer architecture. Complexity is examined in terms of the algorithmic complexity and number of parameters intrinsic of each optimization technique. All of the experiments are carried out on real datasets from medical diagnosis problems. This is an area that has received much attention from the machine learning community and that provides interesting benchmark-problems and recent real problems.

The main contributions of this paper are the following: (i) Different recent and successful techniques are reviewed and compared under strict criteria of optimality on medical problems. (ii) Novel tools are developed to systematically rank these techniques under said criteria and to improve the measure of generalization (PSV) while maintaining effectiveness (Accuracy). (iii) Through comprehensive experimental evaluation the EDAs are identified as the optimal technique.

The remaining sections of this paper are structured as follows: Section II provides a theoretical background about SVMs and optimization strategies. Section III presents the state of the art regarding the hyper-parameters optimization for SVM classifiers. The experimental materials and methods are described in Section IV. Experimental results are reported and discussed in Section V. Finally, conclusions and further directions to extend this research are offered in Section VI.

II. THEORETICAL BACKGROUND

A brief introduction of relevant concepts used in this paper regarding support vector machines (subsection II-A) and optimization strategies (subsection II-B) is provided in this section.

A. SUPPORT VECTOR MACHINES

Support Vector Machines are supervised learning models with many applications including classification, regression and outlier detection [2]. For pattern classification, SVMs solve binary (i.e. two-class) problems by using the formulation provided below [11], [12].

Given a set of training data: $\{\mathbf{z}_i, y_i\}_{i=1}^{N}$, where $\mathbf{z}_i \in \mathbb{R}^D$ is the i-th input vector and $y_i \in \{+1, -1\}$ its corresponding class label, the basic SVM obtains the optimal separating hyperplane:

$$h(\mathbf{z}) = \mathbf{w}^{\mathrm{T}}\mathbf{z} + b \tag{1}$$

where the weight vector $\mathbf{w}$ is defined by a linear combination of relatively few data points called support vectors and b is a scalar known as bias. When data is not linearly-separable, the separating hyperplane is obtained by means of a set of slack variables, $\{\xi_i\}_{i=1}^{N}$ and solving the following quadratic programming problem:

$$\begin{aligned} &\min \quad \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{N}\xi_i \\ &\text{s.t. } y_i(\mathbf{w}^{\mathrm{T}}\mathbf{z}_i + \mathrm{b}) \geq 1 - \xi_i, \quad \xi_i > 0 \quad \text{for } i = 1, \ldots, \mathrm{N} \end{aligned} \tag{2}$$

where C is a scalar parameter called the penalty factor. Introducing the nonnegative Lagrange multipliers λ and following the Karush-Kuhn-Tucker conditions [12], the problem in (2) becomes the following dual problem:

$$\begin{aligned} &\max \quad L(\lambda) = \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \lambda_i \lambda_j K(\mathbf{z}_i, \mathbf{z}_j) \\ &\text{s.t. } C \geq \lambda_{\mathrm{i}} \geq 0 \; \forall i = 1, \ldots, N, \quad \text{and} \quad \sum_{i=1}^{N}\lambda_i y_i = 0 \end{aligned} \tag{3}$$

Given two arbitrary input vectors, $\mathbf{z}$ and $\mathbf{z}'$, the function $K(\mathbf{z}, \mathbf{z}')$ in (3), defined on $\mathbb{R}^D \times \mathbb{R}^D$ is called a kernel if there exists a mapping to the Hilbert space $\phi: \mathbb{R}^D \rightarrow \mathrm{H}$ such that $K(\mathbf{z}, \mathbf{z}') = \langle \phi(\mathbf{z}), \phi(\mathbf{z}') \rangle$ [12]. This latter formulation of the SVM problem is called C-SVM and can be solved through several strategies [13]. The PSV (proportion of vectors $\mathbf{z}_i$ with corresponding $\lambda_i > 0, i = 1, \ldots, N$) has been proved to be a good estimate of the generalization capability of an SVM under the premise that with fewer support vectors, the decision boundary will be less complex, i.e less prone to overfitting [14].

The kernel function can take many different forms; a function that has been widely used since the beginning of the SVM theory is the Radial Basis Function (RBF or Gaussian) kernel, $K_{\mathrm{RBF}}(\mathbf{z}, \mathbf{z}') = \mathrm{e}^{-\gamma \|\mathbf{z} - \mathbf{z}'\|^2}$. The kernel function can add parameters, like the size of the basis function γ, to the SVM formulation that may significantly affect the performance of this classifier. Optimization techniques that can be used to solve the hyper-parameter tuning problem are introduced in the next section.

B. OPTIMIZATION STRATEGIES

In the following exposition, the solution- or search-space is the d-dimensional space conformed by the d parameters that have to be optimized. The search consists of traversing this space while evaluating the objective function for the different parameter configurations (d-dimensional points) and finally reporting the best configuration found.

The simplest technique for exploring the search space is by means of a uniformly spaced grid, this is called Grid Search (GS). The technique is only useful when the search space is composed of very few dimensions; since GS is an exhaustive method and the number of intersections in the grid grows exponentially with the number of dimensions to be explored [15]. Even whenever feasible, GS is not an efficient search strategy. A more efficient method, known as

Random Search (RS) was developed in the 1960s [15]. There are many variations to the original method. Nevertheless, both techniques may be effective in finding a local optimal solution and they are typically employed for simple problems, including hyper-parameter tuning [4]. The advantage of RS over GS is that the former technique may evaluate much fewer points than the latter to achieve the same solution, and thus dimensionality is generally not considered a problem with RS. Whenever a particular probability distribution other than the uniform distribution is used to generate new solutions, the algorithms belong to a new family known as EDAs [16].

EDAs explore the solution space by iteratively building and sampling explicit probabilistic models of candidate solutions that guide their search. The generic procedure is conceptually simple: EDAs start with a population of solutions uniformly drawn from the solution space. Next, these solutions are ranked by means of an objective function related to the problem at hand, and a fraction of the best solutions is selected to construct an estimate of their probability distribution. By sampling this distribution new solutions are generated to update the model. Thus the estimated probability distribution is improved. The process is repeated until a termination criterion is met (usually, when a sufficiently good solution has been found or when a predefined number of iterations have been performed). The pseudocode of the generic EDA is shown in Figure 1.

	General Estimation of Distribution Algorithm
1	Generate initial population of solutions: $X^{(0)} = \{\mathbf{x}_i\}_{i=1}^{n}$ at iteration $t = 0$ uniformly
2	**while** (stopping criteria are not met)
3	Compute objective function of each solution $g\left(X^{(t)}\right)$
4	Select a proportion of the best solutions, $S^{(t)}$ from $X^{(t)}$
5	Build a probabilistic model $P^{(t)}$ based on $S^{(t)}$
6	Sample $P^{(t)}$ to generate new solutions $X'^{(t)}$
7	Substitute / Incorporate $X'^{(t)}$ into $X^{(t)}$
8	Update $t = t + 1$
9	**end while**
10	Output the best solution found, $\mathbf{x}^*$, as the final result

FIGURE 1. Pseudocode of the general EDA.

Hauschild and Pelikan [16] state that "The important step that differentiates EDAs from many other metaheuristics is the construction of the model that attempts to capture the probability distribution of the promising solutions." Depending on the specific probabilistic model built, an EDA takes a particular name. Two of these particular cases of probabilistic models are considered in this work: the Univariate Marginal Distribution Algorithm (UMDA) [17] and the Boltzmann Univariate Marginal Distribution Algorithm (BUMDA) [18]. These are reviewed below.

Univariate marginal distribution algorithms are designed to deal with problems where the components of a solution are independent; in other words, these algorithms are restricted to problems where the solution variables show no or weak correlation between them. The advantages of working under the independence assumption are simplicity and low computational cost. The UMDA builds a Gaussian model from a set of solutions, $S = \{\mathbf{x}_i\}_{i=1}^{M}$ with parameters:

$$\mu_k = \frac{1}{M}\sum_{i=1}^{M} x_{i,k}$$
$$\sigma_k = \left(\frac{1}{M-1}\sum_{i=1}^{M}\left(x_{i,k}-\mu_k\right)^2\right)^{1/2} \quad (4)$$

where M is the number of solutions considered to compute these parameters and $x_{i,k}$ represents the k-th component of solution $\mathbf{x}_i$, that in general is a d-dimensional vector in R^d.

The BUMDA modifies the UMDA by employing a model based on the Boltzmann distribution, which in turn is approximated by a Gaussian model through analytical minimization of the Kullback-Leibler divergence between the two distributions [18]. Mühlenbein and collaborators have shown that Boltzmann distribution-based EDAs converge to the optimum of the objective function [9], [11], [16]. Also, the BUMDA incorporates a truncation selection method which is designed to accelerate convergence as well as to free the user from having to set the number of samples that are selected for parameter estimation. The parameters of the Gaussian distribution used by the BUMDA are computed as follows:

$$\mu_k = \frac{1}{\tilde{g}}\sum_{i=1}^{M} x_{i,k}\, g(\mathbf{x}_i)$$
$$\sigma_k = \left(\frac{1}{\tilde{g}+1}\sum_{i=1}^{M}\left(x_{i,k}-\mu_k\right)^2 g(\mathbf{x}_i)\right)^{1/2} \quad (5)$$

where M is automatically adjusted by a truncation selection method, $g(\mathbf{x}_i)$ is the objective function value of the i-th solution, and $\tilde{g}$ represents the sum of the objective function values over the M selected solutions.

Through recent years, many biology-inspired optimization algorithms have been developed. Within these, a large number of population-based algorithms that model the collective, emerging behavior of multiple biological organisms such as flocks of birds, schools of fish, colonies of ants, swarms of bats, fruit flies, fireflies, etc. have been proposed. The success of these methods has led to an ever increasing interest on a whole new research area, known as swarm intelligence. The pseudocode of the generic swarm intelligence optimization algorithm is shown in Figure 2.

The most recognized representative (with more than 12,000 citations to date) of swarm intelligence algorithms is Particle Swarm Optimization (PSO), developed by Kennedy and Eberhart [19] in 1995 . Presently, there exist at least two dozen variants of the PSO algorithm and many hybridizations of PSO with other swarm intelligence or evolutionary algorithms. In its most basic form, PSO maintains a population of particles, each consisting of a position (a solution to the problem at hand) and a velocity (the adjusting data in Fig. 2) that is used to change the particle's position iteratively.

A global optimum over the population of particles is used to guide the movement of the particles, so that convergence to

General Swarm-Intelligence based algorithm

1 Initialize population: solutions $\mathbf{x}_i$ and corresponding adjusting data $\Delta\mathbf{x}_i$
2 Initialize particular parameters of the algorithm.
3 Compute objective function of each solution $g(\mathbf{x}_i)$.
4 Set global optimum $g^* = \max\ \{g(\mathbf{x}_1), \ldots, g(\mathbf{x}_n)\}$ at iteration $t = 0$, and identify local ($\mathbf{x}_i^*$) / global ($\mathbf{x}^*$) best solution.
5 **while** ($t <$ Max. number of iterations OR other stopping criteria are not met)
6 **for** (each solution $\mathbf{x}_i$)
7 Update adjusting data: $\Delta\mathbf{x}_i{}^{(t+1)}$
8 Generate new solution: $\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \Delta\mathbf{x}_i{}^{(t+1)}$ or by other rule.
9 Evaluate objective function at new locations: $g\left(\mathbf{x}_i^{(t+1)}\right)$
10 [Replace local best solution $\mathbf{x}_i^*$]
11 **end for**
12 Set the current global optimum $g^*$and best solution $\mathbf{x}^*$
13 Update $t = t + 1$
14 **end while**
15 Output the final results $\mathbf{x}^*$ and g^*

FIGURE 2. **Pseudocode of the general swarm intelligence algorithm.**

this optimum is achieved. Meanwhile, a random factor introduced into the particles' velocities ensures that the particles are not trapped in local optima and that solution diversity is preserved throughout the optimization procedure.

III. RELATED WORK

The idea of using metaheuristics to optimize the hyper-parameters of SVM classifiers dates back several years. In 2008, Lin and collaborators reported on the study of simulated annealing [22] and of PSO [9] for hyper-parameter determination and feature selection in SVMs. They compared the metaheuristic methods against grid search on 11 and 17 datasets from the University of California at Irvin (UCI) Machine Learning Repository, respectively. Unfortunately, the conclusion offered from both studies, although presented separately, is virtually the same: the authors conclude in each case that their proposed approach is superior to the other approaches, including a genetic algorithm (GA) approach developed by Huang *et al.* [23], in terms of classification accuracy of SVM classifiers with RBF kernel. The two proposed approaches were not compared to each other.

In 2011, Zhao *et al.* also proposed the use of a GA to simultaneously optimize the feature subset and the hyper-parameters of an SVM classifier with RBF kernel function [5]. The approach was tested on 12 datasets from UCI Machine Learning Repository under a 10-fold cross validation scheme. The GA approach with feature selection managed to produce SVMs with slightly superior classification accuracy than GS over all datasets employed.

In 2015, Zhang and Zhang [24] compared the Social Emotional Algorithm (SEOA) against the PSO algorithm on six datasets from the UCI Repository. Interestingly, they employed the accuracy (Acc.) together with the PSV to guide the optimization process; this was done by minimizing their weighted difference, namely: 0.8Acc. – 0.2PSV. Both of the optimization algorithms found SVMs with very similar accuracy. Also, the average difference in the PSV was only 1.34%. Thus, no significant difference between PSO and SEOA was found. Further, a disadvantage of the SEOA is that it depends on many inter-related parameters, making its use difficult. For this reason, PSO is included in this work.

Also in 2015, Chao and Horng [7] applied the firefly optimization algorithm, FA [20], for tuning of the SVM hyper-parameters with RBF kernel together with the Lagrange multipliers representing the solution of the SVMs. From experiments on ten datasets from the UCI Repository and another medical diagnosis dataset, they found that the FA produced SVMs with higher accuracy than PSO and GS. The FA is included in this work.

Later in 2015, Mantovani *et al.* [4] presented the case in favor of using a simple RS method to adjust the hyper-parameters of SVM classifiers. Based on seventy datasets from the UCI Repository and SVM with RBF kernel, they found RS and even GS to be as effective as other metaheuristic approaches, including PSO, GA and an EDA [25]. It is worth noticing at this point that the effectiveness of the simpler methods such as GS and RS is not under question. As the study of Mantovani *et al.* demonstrates, these algorithms can produce solutions that on average are as good as those found by more sophisticated algorithms. The focus of the present study is on the question of optimality.

Recently, Shen *et al.* [8] tested the Fruit fly Optimization Algorithm (FOA) on four biomedical datasets from UCI Repository. FOA was compared against GS, PSO, GA and Bacterial Foraging Optimization. In experiments with 10-fold cross validation, FOA obtained higher classification accuracy in most of the test datasets. The comparison also showed that FOA required less computational time than its competitors. Because of these recent results, the FOA is included in the experimental comparison presented in this work.

Even more recently, Tharwat *et al.* [6] compared the Bat Algorithm (BA) [26] against PSO, GA and GS for the optimization of SVM hyper-parameters with RBF kernel, also in classification. The method was tested on 9 datasets from the UCI Repository. Experiments with 10-fold cross validation showed that BA achieved the lowest error rates in the vast majority of the test datasets. Due to its theoretical advantages and recent results, BA is included in this work.

Most recently, Padierna *et al.* [10] compared UMDA and BUMDA against GS and RS for the optimization of SVM hyper-parameters. From a comparison on eleven datasets from UCI Repository, they concluded that the EDAs are significantly more efficient (in terms of PFC) than the GS and RS methods, and more importantly, that the individual solutions that were explored by the EDAs are more closely distributed around the optima. Besides the classification accuracy, the PSV was also examined, although it was not used by the algorithms to guide their search of the solution space. UMDA and BUMDA are considered in this work.

TABLE 1. Set of experimental and operational parameters used in previous works.

Shared Parameters	BA [6]	EDAs [10]	FA [7]	FOA [8]	PSO [9]	This work
Population Size (n):	20	50	20	8	6	50
Max. Iteration Number:	20	10	200	250	50	20
Dataset Scaling:	[0,1]	[-1,1]	[-1,1]	[-1,1]	[-1,1]	[-1,1]
Validation Method:	10-fold	10-fold	5-fold	5-fold	10-fold	10-fold
Performance Index:	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
Trials:	1	35	1	10	1	35
Penalty Factor (C):	$(0.01, 35000)$	$(2^{-5}, 2^{5})$	$(2^{-15}, 2^{15})$	$(2^{-5}, 2^{15})$	(0.01,35000)	$(2^{-5}, 2^{5})$
RBF size (γ):	$(0.01, 100)$	$(2^{-5}, 2^{2})$	$(2^{-5}, 2^{5})$	$(2^{-15}, 2)$	(0.01,100)	$(2^{-5}, 2^{2})$
Particular Parameters						
Number of Parameters:	5	0-1	3	2	3	

BA: Initial Loudness $A_0 = 0.5$; Initial Pulse rate $r_0 = 0.5$; $\alpha = 0.9$; $\gamma_{BA} = 0.9$; Frequency: $f \in [0,2]$
FA: Light Absorption Coeff. $\omega = 0.1$; Initial Brightness $\beta_0 = 1$; $\mathbf{u} \sim \mathrm{U}(-0.5, 0.5)$
FOA: Scaling Factor $a = 20$; Offset Term $b = 10$, $\mathbf{u} \sim \mathrm{U}(-1, 1)$
PSO: Cognition Learning Factor $C_1 = 2.0$; Social Learning Factor $C_2 = 2.0$; Inertia weight $w = 1.0$
UMDA: Sample size M = top 25% of population, according to fitness

Experimental Methodology

1 **Input**: User-defined parameters and stopping criteria[1] ($uParams$)
2 **Output:** Performance Information (P.info) to carry out statistical analysis.
3 **for each** experimental trial (i) // i :1 to 35
4 $POP_i^0 \leftarrow$ Generate new initial population of hyper-parameters
5 **for each** $d \in Datasets$
6 Partition $Dataset_d$ into $Folds = \{Training_{k,d}, Test_{k,d}\}_{k=1}^{10}$
7 **for each** $m \in Metaheuristics$
8 P. Info$(m, d, i) \leftarrow Metaheuristic_m(POP_i^0, uParams, LIBSVM, Folds)$
9 **end for**
10 **end for**
11 **end for**

[1]Stopping criteria include: 1. a maximum number of iterations. 2. minimum std. deviation on top 25% best elements of POP_i^t, and 3. maximum number of Function Calls (SVMs evaluated).

FIGURE 3. Pseudocode of the experimental methodology.

IV. MATERIALS AND METHODS

A. EXPERIMENTAL METHODOLOGY

The general process employed to carry out our experiments consists of the steps shown in Figure 3. The function *Metaheuristic* in line 8 performs the hyper-parameter tuning of SVMs with RBF kernel and receives an initial population POP_i^0 per experimental trial. User-defined parameters for the metaheuristics are shown in TABLE 1.

Internally, the individual solutions correspond to SVMs trained with the LIBSVM solver [27] and evaluated by 10-fold cross-validation. In each experimental trial all the metaheuristics are evaluated using the same data folds and initial population to make the comparison between them as fair as possible. Results are stored into P.Info(m, d, i) for each metaheuristic evaluation, these include: accuracy and PSV of the best SVM obtained, and the PFC used by the m-th metaheuristic, on the d-th dataset in the i-th experimental trial. The whole population of solutions is also stored for subsequent analysis.

For the sake of reproducibility, the specific rules for generating individuals of the algorithms in this work are summarized in TABLE 2; their computational complexity, given by the sum of: initialization (A), global best calculation (B), population updating (C) and computation of extra information (D), is also reported. Notice that evaluating a set of d hyper-parameters on an SVM trained with N samples, has a complexity of $O(N^3)$. TABLE 1 shows the values of the optimal parameters reported in previous studies. Based on these, the parameters used for all algorithms in this work were set and are shown in the last column. Prior to any other processing, the data in the test datasets were scaled to the range $[-1, 1]$ to prevent large data values from dominating the solutions. The algorithms were allocated a budget of function calls to find the optimal solution. Some algorithms perform more function evaluations per iteration than others, so the number of function calls used by each algorithm is the more objective measure of efficiency.

An early stopping criterion consisted in testing whether the variance of the top 25% solutions maintained by an algorithm was smaller than a threshold value (the threshold was set to 0.01, stressing the search further than this threshold did not lead to significant improvements in our experiments); if true, this condition signals the convergence of the algorithm. Algorithms that converge earlier are more efficient, although they may be less effective if their best solution found is suboptimal. All experiments were performed on an x3650 M4 IBM server with 16 Intel Xeon CPUs running at 2.6 GHz and 32 GB of RAM. The algorithms were implemented in the Java programming language using multithreading, where each thread executed an independent call of the Metaheuristic function presented in Figure 3.

B. DESCRIPTION OF MEDICAL DIAGNOSIS DATASETS

The experiments were performed on fifteen datasets from the UCI Machine Learning Repository. These datasets represent important disease diagnostic problems and were selected because of their current relevance and because there is still room for improvement in classification performance. All the

TABLE 2. Solution-generating rules of different optimization algorithms.

Updating Rules	Complexity: A+B+C+D
$f_i = f_{min} + (f_{max} - f_{min}) \times rnd$ $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^t + (\mathbf{x}^* - \mathbf{x}_i^t) f_i$	**BA** $\approx O(p\,N^3)$: $O(p\,N^3) + O(p) + G[O(p\,N^3) + O(p)\,] + O(p)$
$\boldsymbol{\mu}_{\mathrm{B}}^t, \boldsymbol{\sigma}_{\mathrm{B}}^t$ are calculated with Eq. (5) $\mathbf{x}_i^{t+1} = Gaussian(\boldsymbol{\mu}_{\mathrm{B}}^t, \boldsymbol{\sigma}_{\mathrm{B}}^t) \approx Boltzmann$	**BUMDA** $\approx O(p\,N^3)$: $O(p\,N^3) + 0 + G\big[O(p\,N^3) + O\big(p\,log(p)\big)\big] + 0$
$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \beta_0 e^{-\omega s_{ij}^2}\big(\mathbf{x}_j^t - \mathbf{x}_i^t\big) + \mathbf{u}_i$ Assuming j-th firefly is better than i-th firefly.	**FA** $\approx O(p^2\,N^3)$: $O(p\,N^3) + 0 + G[O(p^2\,N^3)\,] + 0$
$\mathbf{y}_i = \mathbf{y}_{(\mathbf{x}^*)} + a\mathbf{u}_i - \mathbf{1}b \quad \mathbf{x}_{i,j}^{t+1} = 1/\big(y_{i,j}^2 + z_{i,j}^2\big)^{0.5}$ $\mathbf{z}_i = \mathbf{z}_{(\mathbf{x}^*)} + a\mathbf{u}_i - \mathbf{1}b \quad \big\{\big(y_{i,j}, z_{i,j}\big)\big\}_{j=1}^{d}$ is the fruit-fly space	**FOA** $\approx O(p\,N^3)$: $O(p\,N^3) + O(p) + G[O(p\,N^3) + O(p)\,] + O(p\,d)$
$\mathbf{v}_i^{t+1} = \mathbf{v}_i^t + C_1(\mathbf{x}_i^* - \mathbf{x}_i^t) \times rnd_1 + C_2(\mathbf{x}^* - \mathbf{x}_i^t) \times rnd_2$ $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1}$	**PSO** $\approx O(p\,N^3)$: $O(p\,N^3) + O(p) + G[O(p\,N^3) + O(p)\,] + 0$
$\boldsymbol{\mu}^t, \boldsymbol{\sigma}^t$ are calculated with Eq. (4) $\mathbf{x}_i^{t+1} = Gaussian(\boldsymbol{\mu}^t, \boldsymbol{\sigma}^t)$	**UMDA** $\approx O(p\,N^3)$: $O(p\,N^3) + 0 + G\big[O(p\,N^3) + O\big(p\,log(p)\big)\big] + 0$

$\mathbf{u}, \mathbf{v}, \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^d$; $\mathbf{x}^*$: global best so far, $\mathbf{x}_i^*$: local best (best solution of individual i) so far.
$\mathbf{u}$, rnd: uniformly distributed random vector and number, respectively. $\mathbf{1}$: vector of ones. $s_{ij}^2 = \|\mathbf{x}_i - \mathbf{x}_j\|$.
G: maximum number of iterations, p: population size, d: scale of the problem, N: training samples

TABLE 3. Details of the medical diagnosis datasets employed

Dataset	Instances, Attributes	Optimum Accuracy	Reference(s)
1. BCWD	569, 30	97.62	[28]
2. BCWO	683, 10	96.90 | 97.31 | 99.2	[8] [29] [6]
3. BCWP	194, 33	78.2	[10]
4. DR	1151, 19	74.37	[30]
5. Fertility	100, 10	88	[4]
7. HBCS	306, 3	73.55 | 76.5	[29] [10]
8. Kidney	400, 25	-	-
9. Liver	345, 6	72.45 | 71.6 | 88	[28] [10] [6]
10. Parkinson1	1040, 27	-	-
11. Parkinson2	195, 22	95 | 95.3	[4] [8]
12. PID	768, 8	77.46 | 77.73 | 85.7	[8] [29] [6]
13. SHD	270, 13	83.70 | 85 | 86	[28] [10] [4]
14. TLCS	470, 17	85.3	[4]
15. VC	310, 7	87	[4]
16. VR	126, 309	-	-

datasets are two-class classification problems and correspond to the following medical diagnosis problems (short labels are enclosed in parentheses and the datasets are sorted with respect to these labels): 1. Breast cancer Wisconsin diagnostic (BCWD); 2. Breast cancer Wisconsin original (BCWO); 3. Breast cancer Wisconsin prognostic (BCWP); 4. Diabetic retinopathy from U. of Debrecen (DR); 5. Fertility (Fertility); 6. Haberman's survival after surgery for breast cancer (HBCS); 7. Chronic kidney disease (Kidney); 8. Liver disorders (Liver); 9. Parkinson speech (Parkinson1); 10. Oxford Parkinson's disease detection (Parkinson2); 11. Pima Indians diabetes (PID); 12. Statlog heart disease (SHD); 13. Thoracic surgery survival after surgery for lung cancer (TLCS); 14. Vertebral column (VC); 15. LSVT voice rehabilitation (VR). Most of these datasets have been used in previous works, so that data are available regarding the best classification accuracy reported so far, see TABLE 3.

C. PERFORMANCE INDEX

As mentioned in Section III, Zhang and Zhang [24] combined the PSV with Accuracy to guide the optimization of hyper-parameters. In this work, a new measure of SVM performance is presented as a function of Accuracy and PSV. Provided that these are percentages, the Performance Index for SVM classifiers (PI_{SVM}) is proposed as:

$$\mathrm{PI}_{SVM}(Acc, PSV) = \exp\left(\frac{-\kappa\left(PSV + (100 - Acc)^2\right)}{200 - PSV - Acc + \varepsilon}\right) \tag{6}$$

where $\kappa > 0$ is a constant factor that shapes the function (in this work, $\kappa = 0.1$) and ε is a small number to prevent a division by zero. The PI_{SVM} is bounded in the range [0, 1] and can be used as an alternative to the fitness function based solely on accuracy appearing in the majority of previous works. Notice that, contrary to the idea in [24], the EI_{SVM} does not describe a linear relationship between Accuracy and PSV. The PI_{SVM} is employed to better examine the performance of the algorithms and, in a final experiment, it is tested to lead the optimization of the best algorithm identified.

V. RESULTS AND DISCUSSION

This section is divided into four subsections. The main results are presented in subsection V-A. Statistical analysis is carried out in subsection V-B. Subsection V-C contains detailed performance analysis. Finally, the results regarding performance improvement are reported in subsection V-D.

A. EXPERIMENTAL RESULTS

The results obtained from the experiments described above are summarized in this section. TABLE 4 shows the average classification accuracy over the 35 trials. TABLE 5 shows the corresponding average PSV, used by the SVM classifiers that achieved said accuracies. TABLE 6 contains the average PFC required by the algorithms to produce the reported results of accuracy and PSV. All of these results are presented as percentages, and the standard deviation is provided in brackets.

TABLE 4. Best accuracy as percentage: Avg. over 35 trials (Std. Dev.).

Dataset	BA	BUMDA	FA	FOA	PSO	UMDA
BCWD	98.23	**98.27**	98.25	97.98	98.16	98.26
	(0.21)	(0.17)	(0.18)	(0.22)	(0.23)	(0.17)
BCWO	97.14	97.15	**97.18**	97.10	97.07	97.03
	(0.14)	(0.28)	(0.12)	(0.26)	(0.25)	(0.32)
BCWP	82.12	**82.58**	82.37	80.31	82.25	82.33
	(0.96)	(1.29)	(0.80)	(1.48)	(0.97)	(1.41)
DR	74.30	**74.46**	74.30	73.57	74.26	74.37
	(0.33)	(0.28)	(0.26)	(0.62)	(0.27)	(0.35)
Fertility	**89.19**	**89.19**	**89.19**	**89.19**	**89.19**	**89.19**
	(0.21)	(0.21)	(0.21)	(0.21)	(0.21)	(0.21)
HBCS	74.50	74.54	**74.57**	74.24	74.47	74.51
	(0.53)	(0.50)	(0.56)	(0.54)	(0.59)	(0.55)
Kidney	99.54	99.57	**99.60**	99.23	99.58	99.56
	(0.22)	(0.39)	(0.22)	(0.34)	(0.21)	(0.25)
Liver	73.91	**74.15**	74.04	73.29	73.88	74.04
	(0.58)	(0.49)	(0.56)	(0.60)	(0.57)	(0.46)
Parkinson1	71.94	**72.13**	71.96	71.46	71.89	72.09
	(0.49)	(0.54)	(0.53)	(0.56)	(0.54)	(0.57)
Parkinson2	95.88	95.85	**95.98**	95.74	95.86	95.85
	(0.60)	(0.57)	(0.59)	(0.55)	(0.61)	(0.60)
PID	77.65	**77.98**	77.75	77.25	77.61	77.87
	(0.32)	(0.26)	(0.32)	(0.48)	(0.32)	(0.29)
SHD	84.48	**84.67**	84.50	83.98	84.52	84.54
	(0.54)	(0.37)	(0.46)	(0.37)	(0.50)	(0.49)
TLCS	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**	**85.11**
	(0.00)	(0.05)	(0.04)	(0.00)	(0.00)	(0.04)
VC	85.90	**86.02**	85.94	85.55	85.91	85.95
	(0.50)	(0.46)	(0.41)	(0.52)	(0.51)	(0.49)
VR	89.40	87.67	89.43	85.17	**89.49**	88.68
	(4.04)	(4.97)	(4.03)	(4.94)	(4.04)	(4.00)

TABLE 5. PSV as Percentage: avg. Over 35 Trials (Std. Dev).

Dataset	BA	BUMDA	FA	FOA	PSO	UMDA
BCWD	12.73	**11.95**	12.51	13.51	12.91	12.36
	(2.77)	(1.24)	(2.61)	(3.78)	(2.75)	(1.66)
BCWO	13.31	12.50	**11.33**	16.41	12.94	14.16
	(4.22)	(5.87)	(2.93)	(7.84)	(5.25)	(7.62)
BCWP	57.35	59.24	**56.78**	64.02	56.80	57.64
	(6.39)	(8.74)	(6.37)	(12.5)	(6.55)	(9.04)
DR	65.00	65.40	65.15	**63.99**	65.45	65.37
	(1.53)	(1.09)	(1.23)	(2.50)	(1.62)	(1.41)
Fertility	**94.31**	**94.31**	**94.31**	**94.31**	**94.31**	**94.31**
	(9.26)	(9.26)	(9.26)	(9.26)	(9.26)	(9.26)
HBCS	54.60	**54.34**	54.75	54.71	54.89	54.57
	(1.43)	(1.02)	(1.37)	(1.49)	(1.60)	(1.27)
Kidney	13.47	13.51	**11.77**	20.56	12.30	13.54
	(7.19)	(9.99)	(3.29)	(12.4)	(5.17)	(5.84)
Liver	71.73	71.18	70.77	**69.39**	70.27	70.82
	(4.15)	(3.81)	(3.87)	(2.87)	(3.63)	(3.45)
Parkinson1	73.27	72.79	73.59	73.21	73.08	**72.30**
	(3.18)	(2.39)	(2.90)	(3.65)	(3.28)	(2.54)
Parkinson2	57.67	56.10	**55.61**	57.56	57.63	58.03
	(9.43)	(8.95)	(8.43)	(9.47)	(9.48)	(9.97)
PID	55.96	55.76	54.95	56.66	56.05	**54.94**
	(4.23)	(3.72)	(3.62)	(5.32)	(3.88)	(2.88)
SHD	50.53	45.93	48.51	68.15	51.74	**44.86**
	(5.96)	(4.63)	(4.42)	(20.2)	(8.19)	(4.66)
TLCS	59.03	**58.88**	59.28	59.25	59.32	59.32
	(12.2)	(11.4)	(11.6)	(11.7)	(11.6)	(11.2)
VC	42.45	41.62	42.09	**40.98**	41.34	41.95
	(4.69)	(4.39)	(4.14)	(4.20)	(4.70)	(4.81)
VR	59.28	81.67	**58.00**	82.80	59.85	68.96
	(9.29)	(17.2)	(7.96)	(16.6)	(10.8)	(18.2)

In each case, the best result per dataset is shown in bold typeface. Finally, Tables TABLE 7 and TABLE 8 present the best hyper-parameters found by each of the metaheuristics.

TABLE 6. PFC as percentage: avg. over 35 trials (Std. Dev.).

Dataset	BA	BUMDA	FA	FOA	PSO	UMDA
BCWD	98.7	22.0	100.0	100.0	97.4	**21.7**
	(7.6)	(8.15)	(0.0)	(0.0)	(15.2)	(8.3)
BCWO	69.6	**27.3**	99.0	61.6	69.3	28.7
	(39.8)	(10.1)	(4.4)	(36.8)	(43.2)	(15.5)
BCWP	100.0	51.9	100.0	**10.0**	100.0	41.6
	(0.0)	(19.4)	(0.0)	(0.0)	(0.0)	(14.2)
DR	100.0	34.6	100.0	100.0	100.0	**31.0**
	(0.0)	(11.1)	(0.0)	(0.0)	(0.0)	(9.8)
Fertility	13.0	8.1	38.3	**8.0**	**8.0**	8.3
	(2.5)	(2.73)	(14.4)	(2.5)	(2.5)	(3.2)
HBCS	87.0	52.4	100.0	**10.0**	84.6	38.9
	(29.5)	(23.4)	(0.0)	(0.0)	(34.4)	(16.8)
Kidney	99.0	**21.6**	100.0	100.0	100.0	23.0
	(5.1)	(5.39)	(0.0)	(0.0)	(0.0)	(11.9)
Liver	100.0	55.3	100.0	**10.3**	100.0	49.9
	(0.0)	(23.3)	(0.0)	(1.2)	(0.0)	(21.6)
Parkinson1	100.0	42.4	100.0	100.0	100.0	**38.0**
	(0.0)	(18.2)	(0.0)	(0.0)	(0.0)	(13.0)
Parkinson2	61.0	18.1	90.9	**11.1**	69.0	19.9
	(40.5)	(12.7)	(17.1)	(2.7)	(43.6)	(13.0)
PID	99.7	54.6	100.0	**32.6**	100.0	37.3
	(1.7)	(24.7)	(0.0)	(28.1)	(0.0)	(14.3)
SHD	**50.3**	53.6	100.0	88.6	73.3	54.0
	(37.5)	(19.8)	(0.0)	(29.0)	(40.1)	(21.0)
TLCS	15.0	22.0	99.8	**10.0**	10.1	19.4
	(0.0)	(16.1)	(1.3)	(0.0)	(0.8)	(3.4)
VC	98.0	45.4	100.0	**10.0**	100.0	40.9
	(11.8)	(21.2)	(0.0)	(0.0)	(0.0)	(19.6)
VR	27.6	34.9	97.3	**13.9**	48.7	40.7
	(24.1)	(20.7)	(16.1)	(7.2)	(41.2)	(18.6)

In general, the results in TABLE 4 coincide with those reported in previous works (cf. TABLE 3); the only exception is with respect to the results reported in [6] on datasets Liver and PID where the BA achieved higher accuracies. The corresponding PSV and PFC however, have not been reported before. These are shown in the following tables.

Tables TABLE 4 to TABLE 6 report an average performance of the algorithms. Particularly, TABLE 6 shows the average of SVM evaluations required by each metaheuristic to find the best hyper-parameters. However, average results do not illustrate the complete behavior of the algorithms. For instance, algorithms obtaining the same PFC index may explore the solution space differently. A closer examination reveals the following patterns:

1. All methods except for EDAs use the total function calls allocated (PFC=100) on datasets: DR, BCWD, and Parkinson1.

2. All algorithms obtained similar Accuracy and PSV, but there is a notorious advantage in PFC for EDAs on datasets: Kidney, Parkinson2 and BCWO.

3. All methods obtained good Accuracy and PSV except for FOA; in terms of PFC, EDAs stand out by achieving second and third best places on datasets: SHD and BCWP.

4. All algorithms show small variations in Accuracy and PSV; there is a notorious advantage for FOA in terms of PFC on datasets: Liver, HBCS, PID, and VC.

5. All methods obtained identical Accuracy and similar PSV; the clearest differences between algorithms are in PFC without a clear winner, on datasets: Fertility and TLCS.

TABLE 7. BEST C values: avg. over 35 trials (Std. Dev.).

Dataset	BA	BUMDA	FA	FOA	PSO	UMDA
BCWD	26.34	20.54	26.91	24.06	29.24	17.36
	(1.81)	(3.28)	(2.93)	(5.59)	(3.75)	(3.83)
BCWO	8.86	11.69	15.56	11.77	11.0	12.49
	(7.16)	(4.37)	(8.35)	(9.51)	(2.35)	(7.13)
BCWP	29.42	25.24	23.41	19.2	32.0	22.32
	(3.51)	(0.06)	(4.23)	(8.85)	(0)	(2.5)
DR	30.36	28.42	28.96	27.12	31.72	29.85
	(1.49)	(2.73)	(1.96)	(3.42)	(1.22)	(2.11)
Fertility	0.001	14.98	0.003	0.02	0.001	14.98
	(0)	(10.53)	(0.01)	(0.001)	(0)	(10.53)
HBCS	21.48	11.15	20.52	16.62	25.59	7.92
	(8.76)	(0.22)	(7.13)	(9.12)	(4.49)	(2.78)
Kidney	32.0	23.43	27.38	22.48	31.05	23.45
	(0)	(2.32)	(2.69)	(6.83)	(0.73)	(3.78)
Liver	32.0	30.13	24.57	26.45	31.43	21.62
	(0)	(1.3)	(3.96)	(4.84)	(1.06)	(2.11)
Parkinson1	32.0	23.78	26.43	25.09	32.0	18.74
	(0)	(3.8)	(3.1)	(4.76)	(0)	(4.08)
Parkinson2	23.26	17.0	25.8	21.89	23.16	20.0
	(1.26)	(5.85)	(4.36)	(6.91)	(7.64)	(6.44)
PID	26.24	13.6	22.36	21.18	24.52	7.88
	(4.49)	(3.64)	(7.12)	(7.13)	(3.82)	(0.22)
SHD	27.15	21.92	14.38	12.59	28.6	17.61
	(6.52)	(2.28)	(8.37)	(11.05)	(4.13)	(2.23)
TLCS	0.002	19.02	2.01	0.03	0.001	10.69
	(0.003)	(14.7)	(1.32)	(0.004)	(0)	(10.36)
VC	32.0	28.29	27.98	23.89	29.91	26.31
	(0)	(0.52)	(1.2)	(5.37)	(1.69)	(3.57)
VR	31.75	7.26	28.23	16.25	31.99	11.9
	(0.84)	(3.32)	(1.85)	(10.04)	(0.04)	(7.43)

TABLE 8. Best γ values: avg. over 35 trials (Std. Dev.).

Dataset	BA	BUMDA	FA	FOA	PSO	UMDA
BCWD	0.033	0.045	0.047	0.049	0.042	0.046
	(0.006)	(0.015)	(0.023)	(0.026)	(0.005)	(0.019)
BCWO	0.091	0.123	0.097	0.124	0.093	0.136
	(0.013)	(0.017)	(0.014)	(0.043)	(0.009)	(0.021)
BCWP	0.039	0.049	0.053	0.059	0.05	0.057
	(0.005)	(0)	(0.019)	(0.041)	(0.004)	(0.004)
DR	0.288	0.258	0.346	0.254	0.23	0.316
	(0.097)	(0.128)	(0.096)	(0.127)	(0.119)	(0.108)
Fertility	3.615	0.061	1.894	0.022	2.899	0.061
	(0.457)	(0.073)	(1.879)	(0.002)	(0.715)	(0.073)
HBCS	0.259	0.619	0.454	0.621	0.267	0.707
	(0.022)	(0.005)	(0.266)	(0.272)	(0.075)	(0.16)
Kidney	0.039	0.054	0.043	0.063	0.046	0.053
	(0.008)	(0.009)	(0.007)	(0.027)	(0.007)	(0.009)
Liver	1.106	1.164	1.224	0.859	1.111	1.267
	(0.02)	(0.018)	(0.168)	(0.398)	(0.022)	(0.061)
Parkinson1	0.696	0.996	0.806	0.8	0.725	0.921
	(0.007)	(0.185)	(0.13)	(0.181)	(0.015)	(0.086)
Parkinson2	0.437	0.493	0.408	0.452	0.403	0.485
	(0.013)	(0.073)	(0.093)	(0.138)	(0.087)	(0.124)
PID	0.12	0.081	0.174	0.133	0.049	0.106
	(0.091)	(0.016)	(0.096)	(0.078)	(0.004)	(0.002)
SHD	0.012	0.013	0.016	0.031	0.01	0.017
	(0.008)	(0.001)	(0.007)	(0.021)	(0.005)	(0.002)
TLCS	0.001	0.093	0.065	0.002	0.002	0.09
	(0)	(0.172)	(0.2)	(0)	(0.003)	(0.316)
VC	0.48	0.478	0.861	0.804	0.643	1.009
	(0.007)	(0.005)	(0.383)	(0.445)	(0.2)	(0.456)
VR	0.001	0.006	0.001	0.027	0.001	0.006
	(0)	(0.001)	(0)	(0.017)	(0)	(0.003)

6. Dataset VR could not be included in any of the groups described above.

The best values of hyper-parameter C found by each metaheuristic are shown in TABLE 7. As can be observed, all methods reach similar values for most of the classification problems. The relatively large deviations around the best average values of C indicates that for each problem there exists a range of values for which the best SVM performance was found; in other words, there is a weak influence of the C parameter on the performance of the SVM classifiers. TABLE 8 reports the best values of the hyper-parameter γ found by each metaheuristic. In this table, the opposite effect to the one described for the hyper-parameter C is observed. The small deviations around the average values of γ indicate that this parameter possesses a strong influence on the performance of the SVM classifiers.

B. STATISTICAL ANALYSIS

Following the recommendations of García *et al.* [31] for the comparative evaluation of multiple methods, three different non-parametric statistical tests were performed to test the statistical significance of the differences between the results of the metaheuristics. These tests are: the Friedman test (differences between each pair of index values are equally weighted independently of their magnitude); the Aligned Friedman test (problems are equally weighted regardless of their difficulty, a general behavior of all datasets as a group can be observed); and the Quade test (rankings are weighted based on the difficulty of each problem) [31]. The average rankings, according to each of these statistical tests, are presented in TABLE 9. The corresponding p-values are also shown in this table and when a value is smaller than the significance level of 0.05, it is shown in bold typeface.

Results in TABLE 9 demonstrate that there exist significant differences between at least two methods under all performance indexes. To identify which methods are different from each other, post hoc tests were carried out. It was found that, with statistical significance:

1. In Accuracy: BUMDA is better than all other algorithms, except for FA (F. test). FA is better than FOA and PSO (A-F. test). BUMDA is better than FOA and BA (Q. test).

2. In PSV: FA is better than FOA and UMDA (A-F. test). FA is better than FOA (Q. test).

3. In PFC: UMDA is better than FA, PSO and BA (F. test). UMDA is better than all other methods, except for BUMDA (A-F. test). FOA is better than FA and PSO (Q. test).

To derive the main conclusion from the previous statistical results, notice the following two facts: 1. the FA and BUMDA methods compete for the first two places in terms of Accuracy and PSV while being statistically equivalent according to post hoc tests. 2. UMDA and FOA obtained the best two places in terms of PFC, but none of them is statistically different to BUMDA according to post hoc test. From these two observations it follows that under these evaluation criteria, BUMDA is the best method, since it is as effective as FA, PSO and BA while remaining as efficient as UMDA and FOA.

TABLE 9. Average Ranking and *p*-Values with the RBF kernel.

	Accuracy			PSV			PFC		
	F. test	A-F. test	Q. test	F. test	A-F. test	Q. test	F. test	A-F. test	Q. test
BA	3.933	45.40	4.050	3.800	44.60	3.842	3.933	57.26	4.033
BUMDA	**1.933**	32.23	**1.823**	2.966	41.53	3.279	2.533	25.13	2.750
FA	2.500	**30.33**	2.533	**2.900**	**34.50**	**2.263**	5.500	76.83	5.521
FOA	5.566	79.90	5.895	4.333	62.83	4.808	2.467	31.53	**1.933**
PSO	3.966	51.10	3.880	3.600	43.57	3.342	4.233	59.97	4.412
UMDA	3.100	34.03	2.812	3.400	45.97	3.467	**2.333**	**22.27**	2.350
***p*-value**	**1.18E-6**	**0.033**	**2.96E-10**	0.286	**0.025**	**0.0035**	**1.75E-6**	**0.026**	**3.8E-9**

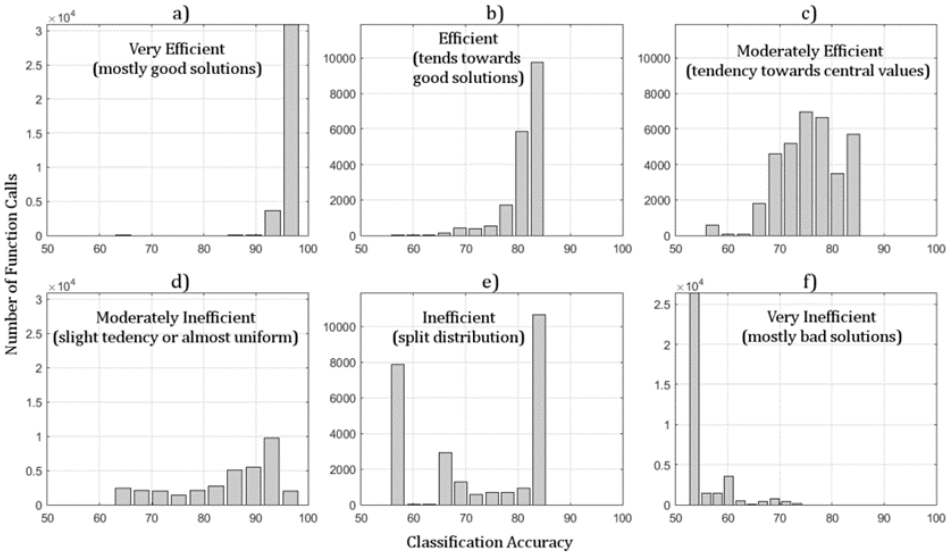

FIGURE 4. Prototype frequency distributions of function calls over accuracy.

C. PERFORMANCE ANALYSIS

To carry out a detailed performance analysis, the complete set of solutions evaluated over all iterations and 35 trials are examined. The objective is to determine the way in which each of the algorithms explores the solution space. This will show which of the algorithms makes better use of the allocated budget of solutions, for instance, by not wasting effort in the evaluation of bad solutions. With six different algorithms and fifteen datasets, there are ninety frequency distributions to be examined. For the sake of clarity and due to space limitations, these ninety distributions have been classified into one of six different prototypes shown in Figure 4. The behavior of the algorithms is analyzed with respect to these prototypes, labelled as Type-A to Type-F distributions (Fig. 4-a to 4-f, respectively).

Type-A distributions (Fig. 4a) correspond to very efficient algorithms since a vast majority of the solutions obtained the highest classification accuracy. Type-B distributions (Fig. 4b) correspond to efficient algorithms, where increasing number of solutions obtained ever higher accuracy. Type-C distributions (Fig. 4c) show a tendency towards the central values. These correspond to moderately-efficient algorithms, because many function calls were used to evaluate solutions that are not too good. Type-D distributions (Fig. 4d) correspond to moderately-inefficient algorithms, because the same amount of function calls were used to evaluate any kind of solution. Type-E distributions (Fig. 4e) correspond to inefficient algorithms that evaluate almost as many bad solutions as medium-and-good solutions taken together. Type-F distributions (Fig. 4f) correspond to very inefficient algorithms that use a vast majority of their function calls to evaluate bad solutions; these algorithms may reach an optimal solution only exceptionally.

Based on these prototype distributions, the following ranking of the algorithms is presented:

1. BUMDA is very efficient, with distributions of type: A=5, B=7, D=2 and E=1.

2. UMDA is very efficient, with distributions of type: A=4, B=8, D=2 and E=1.

3. BA is an efficient algorithm, showing distributions of type: A=4, B=7, D=2, E=1 and F=1.

4. FA is moderately efficient, with distributions of type: A=3, B=5, C=3, E=3 and F=1. This algorithm typically uses many more function calls than the rest, cf. TABLE 6.

5. PSO is an inefficient algorithm, showing distributions of type: A=2, D=4, E=8 and F=1.

6. FOA is very inefficient, with distributions of type: A=2, B=1, D=4, E=1 and F=7.

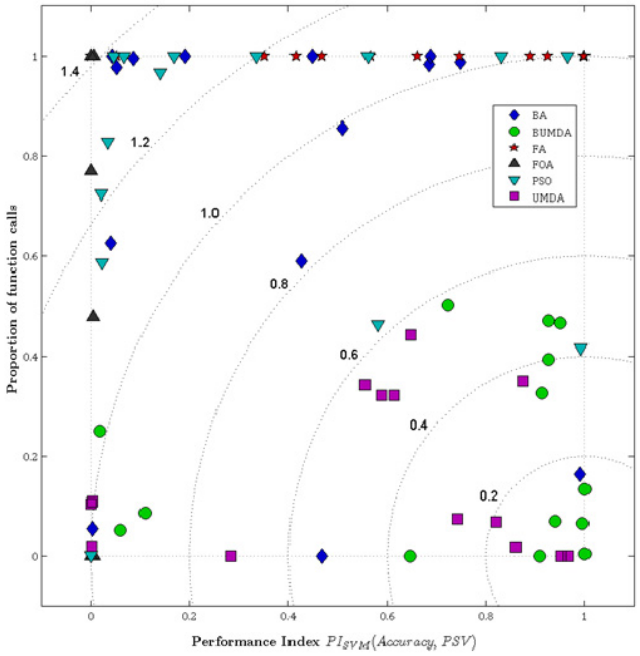

FIGURE 5. Plot of performance index vs PFC.

In order to visualize the observations described in this section, Figure 5 presents the plot of the normalized PI_{SVM} defined in (6) against normalized PFC of the different metaheuristics, per dataset (so there are 15 points for each algorithm). This plot is a valuable visualization tool that allows one to directly compare the different metaheuristics. In this plot, the best algorithm has $PI_{SVM} = 1$ and PFC=0, corresponding to the right-bottom corner of the unit-square depicted. Circles with different radii around that point have been drawn, to represent distances from the optimum.

It can be observed that BUMDA has 5 instances within 0.2 units from the optimum, 3 more instances within 0.4, and other 3 more instances within 0.6 of the optimum. UMDA is similar, but not superior, with 4, 2 and 4 instances within 0.2, 0.4 and 0.6 from the optimum, respectively. In contrast, neither FA nor FOA have any of their instances within these ranges. In summary, this plot clearly shows that both EDAs are significantly superior to all other algorithms considered.

D. PERFORMANCE IMPROVEMENT

Besides performance visualization, the PI_{SVM} can also be used to guide the optimization process itself, which means that the algorithms will search for solutions that optimize classification accuracy as well as PSV simultaneously. To illustrate this, one extra experiment was performed with the BUMDA algorithm (the best ranked algorithm so far), to compare the solutions found with accuracy-based fitness against the solutions with PI_{SVM} used as the fitness function. Results are reported in TABLE 10. The best results are shown in bold typeface.

TABLE 10. Performance as percentage of BUMDA and BUMDA using PI_{SVM}: average over 35 trials (Std. Dev.).

	Accuracy		PSV		PFC	
Dataset	BUMDA vs BUMDA+PI_{SVM}		BUMDA vs BUMDA+PI_{SVM}		BUMDA vs BUMDA+PI_{SVM}	
BCWD	**98.27**	98.15	11.95	**10.22**	22.0	**16.3**
	(0.17)	(0.21)	(1.24)	(0)	(8.15)	(3.5)
BCWO	97.15	**97.16**	12.50	**8.42**	**27.3**	**19.4**
	(0.28)	(0.17)	(5.87)	(0)	(10.1)	(2.91)
BCWP	82.58	**82.65**	59.24	**53.14**	51.9	61.6
	(1.29)	(0.98)	(8.74)	(0.02)	(19.4)	(17.0)
DR	**74.46**	74.11	65.40	**62.05**	34.6	54.6
	(0.28)	(0.47)	(1.09)	(0.01)	(11.1)	(16.0)
Fertility	**89.19**	88.22	94.31	**31.94**	8.1	45.0
	(0.21)	(0.88)	(9.26)	(0.06)	(2.73)	(19.3)
HBCS	74.54	**74.71**	54.34	**53.90**	52.4	53.0
	(0.50)	(0.67)	(1.02)	(0.01)	(23.4)	(22.4)
Kidney	**99.57**	99.46	13.51	**8.39**	**21.6**	**16.9**
	(0.39)	(0.27)	(9.99)	(0)	(5.39)	(3.23)
Liver	**74.15**	73.55	71.18	**65.44**	55.3	59.0
	(0.49)	(0.93)	(3.81)	(0.01)	(23.3)	(19.7)
Parkinson1	**72.13**	71.51	72.79	**66.34**	42.4	50.1
	(0.54)	(0.65)	(2.39)	(0.01)	(18.2)	(17.1)
Parkinson2	**95.85**	94.77	56.10	**34.11**	18.1	40.3
	(0.57)	(0.66)	(8.95)	(0.02)	(12.7)	(18.5)
PID	**77.98**	77.68	55.76	**51.85**	54.6	59.4
	(0.26)	(0.46)	(3.72)	(0.01)	(24.7)	(15.6)
SHD	**84.67**	84.17	45.93	**41.09**	53.6	58.4
	(0.37)	(0.70)	(4.63)	(0.01)	(19.8)	(24.1)
TLCS	85.11	**85.17**	58.88	**38.77**	22.0	36.4
	(0.05)	(0.15)	(11.4)	(0.04)	(16.1)	(18.5)
VC	**86.02**	85.83	41.62	**37.03**	45.4	47.0
	(0.46)	(0.63)	(4.39)	(0.02)	(21.2)	(23.9)
VR	**87.67**	86.39	81.67	**64.37**	34.9	57.0
	(4.97)	(6.89)	(17.2)	(0.16)	(20.7)	(24.7)
Average Difference	0.38 in favor of BUMDA		11.20 in favor of BUMDA+PI_{SVM}		8.68 in favor of BUMDA	

From TABLE 10 it can be observed that the PI_{SVM} led BUMDA to obtain solutions with notably lower PSV. This is achieved without a loss in Accuracy, although a price on the efficiency is paid (reflected by higher PFC). As can be seen, optimality may be defined under different, often opposing criteria; in this case, a trade-off between generalization and efficiency must be made. However, in the context of medical diagnosis, higher performance (effectivity + generalization) is preferred over efficiency of the optimization process. As discussed in Section III, a previously proposed objective function combining Accuracy and PSV [24] did not lead to any substantial improvement; in contrast, the proposed PI_{SVM} leads to noticeable improvement, probably due to its nonlinear nature. Therefore, based on these arguments, the use of the PI_{SVM} is recommended. Up to this point, the considered metaheuristics have been evaluated under six different measures. The final ranking of the algorithms under each measure are presented in TABLE 11.

TABLE 11. Final ranking of the Metaheuristics under optimality criteria.

Criterion	BA	BUMDA	FA	FOA	PSO	UMDA
Effectiveness-Accuracy	4	1	2	6	5	3
Generalization-PSV	5	2	1	6	3	4
Efficiency-PFC	4	3	6	2	5	1
Efficiency-Exploration	3	1.5	4	6	5	1.5
Complexity-Algorithmic	3	3	6	3	3	3
Complexity-Parameters	6	1	4.5	3	4.5	2
Average Ranking	4.20	1.92	3.92	4.33	4.25	2.42

The Effectiveness-Accuracy criterion refers to results in TABLE 4, Generalization-PSV to results in TABLE 5 and Efficiency-PFC to those in TABLE 6. The ranking of these three criteria coincides with the ranking from the statistical analysis summarized in TABLE 9. Efficiency-Exploration criterion relates to the analysis presented in subsection V-C. The Complexity-Parameters and Complexity-Algorithmic criteria were ranked according to the information in TABLE 1 and TABLE 2, respectively. Based on these criteria a final ranking of the algorithms is presented. The conclusion is that the EDAs are the optimal strategy while the rest of the algorithms are grouped together in a lower category.

VI. CONCLUSIONS AND FUTURE WORK

The most important conclusion is that, supported by our experimental results and under the established criteria, the EDAs are the optimal methods for hyper-parameter tuning of SVM classifiers. With respect to the rest of the algorithms, it is important to consider that their performance is dependent on the specific values of their user-defined parameters. Despite using the best reported parameters, these algorithms were ranked significantly lower than those with one or no user-defined parameter. In addition, the use of an optimization method that requires the setting of more parameters than the problem dimensionality should be avoided. Particular conclusions about the bio-inspired metaheuristics are:

BA realized the best exploration among the bio-inspired metaheuristics. It also obtained average ranking under algorithmic complexity and PFC. Its main disadvantage is the large number of user-defined parameters; these make it difficult to use and can negatively affect its performance. FA obtained the lowest place in PFC and algorithmic complexity; this is balanced out by its good performance in Accuracy and PSV. In other words, FA is an algorithm that achieves very good solutions, but it employs far many SVM evaluations, as reflected by its ranking under the exploration criterion. FOA placed second best according to PFC; this apparent efficiency is linked to the lowest ranking in Accuracy, PSV and Exploration. Despite favorable results reported in previous works, FOA appears to be very sensitive to its own parameters. PSO obtained low performance indexes in general. In the performance analysis, it was found that many of the solutions explored by PSO were centered on local optima. This may be due to PSO keeping a local best for each particle, which is not done by the other methods.

In all the considered datasets, most of the algorithms were able to reproduce the results reported in previous works. There are three diagnostic problems that had not been evaluated previously: Chronic kidney disease (Kidney), Parkinson speech (Parkinson1) and LSVT voice rehabilitation (VR). Our results show that Kidney is an easily solvable classification problem, but VR and Parkinson1 require more effort. We consider that the experimental methodology followed in this work is more reliable than those in previous studies because a larger number of experimental trials are carried out under several relevant performance criteria, while other works have focused only on classification accuracy and sometimes PSV. In addition, optimization methods not included in this work can also be compared according to the proposed criteria and evaluation methodology.

It was shown that the proposed performance index (PI_{SVM}) provides two advantages over the Accuracy measure. First, the plot of PI_{SVM} vs efficiency (in this work measured by PFC) becomes a visualization tool that allows the direct comparison of the algorithms in terms of efficiency, effectiveness and generalization, simultaneously. A more important advantage is that through the PI_{SVM}, better solutions (with lower PSV) were found, thus demonstrating that the proposed index leads to a better exploration of the solution space. In this work, the PI_{SVM} was tested with the best ranked algorithm; in the future, a more extensive evaluation of this performance index will be conducted. Also, considering the success of the methods on medical diagnosis problems, a natural extension of this work is to test our findings on other important applications. Finally, the criteria used to evaluate the optimality of hyper-parameter-tuning methods can be applied in other problem domains, such as SVM regression, SVM density estimation, etc.

REFERENCES

[1] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge Univ. Press, 2004.
[2] V. Vapnik, *Statistical Learning Theory*. New York, NY, USA: Wiley, 1998.
[3] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 281–305, 2012.
[4] R. G. Mantovani, A. L. D. Rossi, J. Vanschoren, and A. C. P. L. F. de Bischl, "Effectiveness of random search in SVM hyper-parameter tuning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2015, pp. 1–8.
[5] M. Zhao, C. Fu, L. Ji, K. Tang, and M. Zhou, "Feature selection and parameter optimization for support vector machines: A new approach based on genetic algorithm with feature chromosomes," *Expert Syst. Appl.*, vol. 38, no. 5, pp. 5197–5204, 2011.
[6] A. Tharwat, A. E. Hassanien, and B. E. Elnaghi, "A BA-based algorithm for parameter optimization of support vector machine," *Pattern Recognit. Lett.*, vol. 93, pp. 1–3, Oct. 2016.

[7] C.-F. Chao and M.-H. Horng, "The construction of support vector machine classifier using the firefly algorithm," *Comput. Intell. Neurosci.*, vol. 2015, Jan. 2015, Art. no. 212719.
[8] L. Shen *et al.*, "Evolving support vector machines using fruit fly optimization for medical data classification," *Knowl.-Based Syst.*, vol. 96, no. 15, pp. 61–75, Mar. 2016.
[9] S.-W. Lin, K.-C. Ying, S.-C. Chen, and Z.-J. Lee, "Particle swarm optimization for parameter determination and feature selection of support vector machines," *Expert Syst. Appl.*, vol. 35, no. 4, pp. 1817–1824, 2008.
[10] L. C. Padierna, J. M. Carpio, A. Rojas, H. Puga, R. Baltazar, and H. Fraire, "Hyper-parameter tuning for support vector machines by estimation of distribution algorithms," in *Nature-Inspired Design of Hybrid Intelligent Systems* (Studies, in Computational Intelligence), vol. 667, P. Melin, O. Castillo, and J. Kacprzyk, Eds. Cham, Switzerland: Springer, 2017, pp. 787–800.
[11] A. Shigeo, *Support Vector Machines for Pattern Classification*. New York, NY, USA: Springer, 2010.
[12] N. Deng, Y. Tian, and C. Zhang, *Support Vector Machines*, Boca Raton, FL, USA: CRC Press, 2013.
[13] J. Shawe-Taylor and S. Sun, "A review of optimization methodologies in support vector machines," *Neurocomputing*, vol. 74, no. 17, pp. 3609–3618, 2011.
[14] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 131–159, 2002.
[15] R. E. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton, NJ, USA: Princeton Univ. Press, 2015.
[16] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 111–128, 2011.
[17] H. Mühlenbein, "The equation for response to selection and its use for prediction," *Evol. Comput.*, vol. 5, no. 3, pp. 303–346, 1997.
[18] S. I. Valdez, A. Hernández, and S. Botello, "A Boltzmann based estimation of distribution algorithm," *Inf. Sci.*, vol. 236, pp. 126–137, Jul. 2013.
[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, WA, USA, Nov./Dec. 1995, pp. 1942–1948.
[20] X.-S. Yang, C. Zhihua, X. Renbin, H. G. Amir, and K. Mehmet, Eds., *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*. Oxford, U.K.: Newnes, 2013.
[21] I. Fister, Jr., X.-S. Yang, I. Fister, J. Brest, and D. Fister. (2013). "A brief review of nature-inspired algorithms for optimization." [Online]. Available: https://arxiv.org/abs/1307.4186
[22] S.-W. Lin, Z.-J. Lee, S.-C. Chen, and T.-Y. Tseng, "Parameter determination of support vector machine and feature selection using simulated annealing approach," *Appl. Soft Comput.*, vol. 8, no. 4, pp. 1505–1512, 2008.
[23] C.-L. Huang, M.-C. Chen, and C.-J. Wang, "Credit scoring with a data mining approach based on support vector machines," *Expert Syst. Appl.*, vol. 33, no. 4, pp. 847–856, 2007.
[24] Y. Zhang and P. Zhang, "Machine training and parameter settings with social emotional optimization algorithm for support vector machine," *Pattern Recognit. Lett.*, vol. 54, pp. 36–42, Mar. 2015.
[25] M. Soto, A. Ochoa, and R. J. Arderi, "Gaussian copula estimation of distribution algorithm," Inst. Cybern., Math. Phys., Cuba, Tech. Rep. ICIMAF 2007-406, Jun. 2007.
[26] X. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, 2012.
[27] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[28] L. Sun, K.-A. Toh, and Z. Lin, "A center sliding Bayesian binary classifier adopting orthogonal polynomials," *Pattern Recognit.*, vol. 48, no. 6, pp. 2013–2028, 2015.
[29] A. L. Chau, X. Li, and W. Yu, "Support vector machine classification for large datasets using decision tree and Fisher linear discriminant," *Future Generat. Comput. Syst.*, vol. 36, pp. 57–65, Jul. 2014.
[30] A. Goel and K. S. Saurabh, "Role of kernel parameters in performance evaluation of SVM," in *Proc. 2nd Int. Conf. Comput. Intell. Commun. Technol. (CICT)*, Feb. 2016, pp. 166–169.
[31] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, 2010.

ALFONSO ROJAS-DOMÍNGUEZ was born in Mexico City, Mexico, in 1978. He received the B.Sc. degree in telecommunications engineering from the National Autonomous University of Mexico in 2001, the M.Sc. degree in intelligence engineering in 2003, and the Ph.D. degree in electronics engineering from the University of Liverpool, U.K., in 2007.

From 2008 to 2014, he was a Researcher with recognized institutions, such as the Institute of Biotechnology, from 2008 to 2011, the Center for Research in Mathematics, from 2012 to 2013, and the Center for Research and Advanced Studies, from 2013 to 2014. Since 2014, he has been a Research Fellow with the National Council of Science and Technology (CONACYT) of Mexico. He is currently associated with the Instituto Tecnológico de León, Gto., Mexico.

Dr. Rojas-Domínguez has authored research papers with over 400 citations to date. He has been a member of the National Researchers System from 2008 to 2013 and from 2014 to 2016. He is currently a CONACYT Research Fellow. His main research interests are machine learning, pattern recognition, and computer vision. His current research interests include SVMs, evolutionary algorithms, and deep learning.

LUIS CARLOS PADIERNA was born in Dolores Hidalgo Guanajuato, México, in 1985. He received the B.Eng. degree in computer systems engineering from the Technology Institute of Celaya, Mexico, in 2009, the M.Sc. degree in computer science from the Instituto Tecnológico de León, in 2011, where he is currently pursuing the Ph.D. degree in computer science.

He has authored research papers in the designing of support vector machines. His main research interests are machine learning, pattern recognition, evolutionary algorithms, and deep learning.

JUAN MARTÍN CARPIO VALADEZ was born in León, Gto., México, in 1961. He received the B.Sc. degree in mathematics from the Autonomous University of Nuevo Leon, México, in 1985, and the M.Sc. and Ph.D. degrees in physics (optics) from the University of Guanajuato, México, in 1987 and 1995, respectively.

Since 1994, he has been associated with the Research Division of the Instituto Tecnológico de León, Guanajuato, Mexico.

Dr. Carpio has been a member of the National Researchers System of the National Council of Science and Technology of Mexico, since 1992. His main research interests are optic metrology and data fitting with functions and orthogonal polynomials, intelligent systems, in particular, optimization with heuristics, metaheuristic and hyper-heuristics. His current research interests include SVMs and evolutionary algorithms.

HECTOR J. PUGA-SOBERANES was born in Mexico City in 1960. He received the B.Sc. degree in physics and mathematics from the National Polytechnic Institute, in 1993, and the M.Sc. and Ph.D. degrees in physics (optics) from the University of Guanajuato, México, in 1995 and 2002, respectively.

Since 2002, he has been associated with the Research Division of the Technology Institute of León, Gto., Mexico.

Dr. Puga has been a member of the National Researchers System of the National Council of Science and Technology of Mexico, since 2004. His main research interests are optic metrology and intelligent systems, in particular, optimization whit heuristics, metaheuristics and hyper-heuristics. His current researches include SVMs and evolutionary algorithms.

HÉCTOR J. FRAIRE received the B.S. Math and M.I.S. degrees from the Universidad Autónoma de Nuevo León, México, in 1976 and 1988, respectively, and the Ph.D. degree in computer science from the Centro Nacional de Investigación y Desarrollo Tecnológico, in 2005. He is currently a Professor with the Instituto Tecnológico de Ciudad Madero, México. His scientific interests include metaheuristic optimization and machine learning.