Applied Soft Computing 13 (2013) 3494–3504

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients

Susana M. Vieira[a,*], Luís F. Mendonça[a,b], Gonçalo J. Farinha[a], João M.C. Sousa[a]

[a] Technical University of Lisbon, Instituto Superior Técnico, Department of Mechanical Engineering, Center of Intelligent Systems, IDMEC Av. Rovisco Pais, 1049-001 Lisbon, Portugal
[b] Escola Superior Náutica Infante D. Henrique, Department of Marine Engineering, Lisbon, Portugal

ARTICLE INFO

Article history:
Received 21 September 2012
Received in revised form 18 January 2013
Accepted 8 March 2013
Available online 23 April 2013

Keywords:
Feature selection
Wrapper methods
Particle swarm optimization
Premature convergence
Sepsis
Support vector machines

ABSTRACT

This paper proposes a modified binary particle swarm optimization (MBPSO) method for feature selection with the simultaneous optimization of SVM kernel parameter setting, applied to mortality prediction in septic patients. An enhanced version of binary particle swarm optimization, designed to cope with premature convergence of the BPSO algorithm is proposed. MBPSO control the swarm variability using the velocity and the similarity between best swarm solutions. This paper uses support vector machines in a wrapper approach, where the kernel parameters are optimized at the same time. The approach is applied to predict the outcome (survived or deceased) of patients with septic shock. Further, MBPSO is tested in several benchmark datasets and is compared with other PSO based algorithms and genetic algorithms (GA). The experimental results showed that the proposed approach can correctly select the discriminating input features and also achieve high classification accuracy, specially when compared to other PSO based algorithms. When compared to GA, MBPSO is similar in terms of accuracy, but the subset solutions have less selected features.

1. Introduction

Advances in computer science and acquisition systems have made possible to easily collect and store large databases containing long time series. These databases have become increasingly frequent in distinct fields, including astronomy [1], molecular biology [2], finance [3], marketing [4] and health care [5], and are often used for purposes of data mining and knowledge discovery.

Knowledge is only valuable when it can be used efficiently and effectively. Therefore, extensive research has been made in the urge to find new computational theories and tools, that can aid to the extraction of useful information (knowledge) from these rapidly growing databases. The field of science concerned with automated knowledge discovery is called knowledge discovery in databases (KDD) [6].

Mathematical modeling is the description of a system using mathematical language. For complex and partly understood systems, nonlinear models based on artificial intelligence techniques can be used. Modeling may help in medical diagnosis; a good example is the prediction of sepsis outcome using learning machines. An usual drawback, encountered when modeling real systems, is noise and redundancy in data. Hence, it is very important, when preprocessing data, to choose the optimal feature subset.

Feature selection (FS) or variable selection, is the technique of selecting a subset of relevant features (variables) for building more robust learning models. It not only reduces the processing cost, but also improves the model built from the selected data [7,8]. Thus, its use in healthcare modeling problems helps to prevent medical complications and even patient death. In feature selection the goals are to maximize the model performance and to minimize the number of used features. The solution is dependent on the chosen combination of features, making the FS problem NP-Hard. Metaheuristics such as particle swarm optimization (PSO) [9], evolutionary algorithms (EA) [10], or ant colony optimization (ACO) [11,12] have shown to be well suited for this kind of problems, due to their randomized nature [13]. They are able to find good solutions, without having to try all possible combinations.

The medical condition studied in this paper is sepsis, a common clinical condition defined by a whole-body inflammatory state, called systemic inflammatory response syndrome (SIRS). This clinical condition has different degrees of severity that can lead to severe sepsis and later to septic shock. A patient is considered to be in septic shock if he/she has sepsis associated with arterial hypotension despite "adequate" fluid resuscitation [14]. This advanced stage of sepsis carries a high burden, which translates into a high mortality rate (about 50%) and high costs of treatments, when compared with other intensive care unit (ICU) patients [15,16]. It is now considered

* Corresponding author. Tel.: +351 218419117.
E-mail address: susana.vieira@ist.utl.pt (S.M. Vieira).

1568-4946/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2013.03.021

to be the most common cause of death in noncoronary critical care units. Approximately 150,000 persons die annually in Europe and more than 200,000 in the United States [17].

This paper proposes a modified binary particle swarm optimization (MBPSO) method for feature selection with the simultaneous optimization of SVM kernel parameter setting, applied to mortality prediction in septic patients. An enhanced version of binary particle swarm optimization, designed to cope with premature convergence of the BPSO algorithm is proposed. MBPSO control the swarm variability using the velocity and the similarity between best swarm solutions. This paper uses support vector machines in a wrapper approach, where the kernel parameters are optimized at the same time. The MBPSO is used as a wrapper method, that is solely a feature selection method in which every candidate solution is evaluated using a learning machine [18], such as neural networks (NN), support vector machines (SVM) or fuzzy modeling (FM) [19]. The chosen classification technique, in this case, is support vector machines, once this method has universal function approximation properties [20]. Further, SVM are widely used in computational biology due to their high accuracy, their ability to deal with high-dimensional and large databases, and their flexibility in modeling diverse sources of data [21]. These characteristics are suitable to the problem proposed in this paper. Dealing with clinical data can be problematic, since the available data is usually high-dimensional and very heterogeneous, which means dealing with a large number of features and different types of data. In clinical databases, which is the environment where it is intended to apply the developed method, usually the higher the number of studied features the less the number of patients and samples available, once not all the patients have the same physiological variables registered. This means that the used number of features is always a trade-off between the number of features and the number of patients, which typically translates into tens of features.

We will start by introducing the problem formulation for feature selection in Section 2, where the main concepts of support vector machines are also presented. Then, the classical binary PSO is presented in Section 3. In Section 4, the proposed MBPSO approach is described and its advantages are assessed using a group of benchmark databases. In Section 5, we will apply the MBPSO algorithm to the sepsis outcome prediction problem, and the conclusions are presented in Section 6.

2. Feature selection

The KDD process comprises a series of steps to extract knowledge from data. The first step is *selection* and it consists of acquiring the most useful target data from the available databases. The target database has to be adequately chosen so that it contains sufficient information regarding the system we want to describe. The next two steps (feature construction and feature selection), are part of the feature extraction process and are used with the purpose of extracting the most relevant features of the target data. *Feature construction* (also called data preprocessing) [22], comprehends all the methods that involve some degree of modification to the original feature, e.g. data *standardization*, *normalization* and *noise filtering*. The objective of this crucial preprocessing step is to make the underlying information in data easier to identify. In opposition, *feature selection* does not induce a transformation to the features, it simply searches for the optimal feature subset discarding the features with lowest informative potential.

There is a large number of available feature selection techniques, but there are three aspects that roughly differentiate them [22]:

- feature subset generation (or search strategy);
- evaluation criterion definition (e.g. relevance index or predictive performance);
- evaluation criterion estimation (or assessment method).

The first refers to the applied search strategy to evaluate the solutions in the space of possible feature combinations. The last two correspond to the evaluation criterion, i.e. the method and measures used to assess the quality of each feature subset. Based on the subset evaluation procedure we may divide feature selection algorithms into three classes, *wrapper methods*, *embedded methods* and *filter methods* [23].

Wrappers use a search algorithm to search through the space of possible features and evaluate each subset by running a model on the subset. Wrappers can be computationally expensive and have a risk of over-fitting the model. Filters are similar to wrappers in the search approach, but instead of evaluating against a model, the features are selected by evaluating a performance measure that does not require building a model.

In embedded feature selection methods, similarly to wrapper methods, feature selection is linked to the classification stage. This link is in this case much stronger, as feature selection in embedded methods is included into the classifier construction. Recursive partitioning methods for decision trees such as ID3, C4.5 and CART are examples of such methods.

The main advantage of the wrapper method over embedded methods is a better coverage of the search space. And the main advantage of the wrapper method over filter methods is that in wrappers the predictive performance of the final selected subset is correlated with the chosen relevance measure, or in this case the classifier. When the objective is to obtain a model as accurate as possible and the time to obtain it is not an issue, then the wrapper method is an advantageous choice.

2.1. Wrapper methods

The main characteristic of wrapper methodologies is the use of the predictor as part of the selection procedure. They use a learning machine to score the subsets according to their predictive performance [22]. Wrappers are constituted by three main components:

1. Learning machine;
2. Feature evaluation criteria;
3. Search method.

Wrapper approaches are aimed to improve the results of the specific predictors they work with. Nevertheless, wrapper methods have the associated problem of having to train a classifier for each tested feature subset. This means testing all the possible combinations of features will be virtually impossible and a search method is imperative. During the search, subsets are evaluated without incorporating knowledge about the specific structure of the classification or regression function [22].

Many popular search approaches use greedy hill climbing, which iteratively evaluates a candidate subset of features, then modifies the subset and determines whether or not the new subset is an improvement over the old. Evaluation of the subsets requires a scoring metric that grades a subset of features. In the case of wrapper methods, the feature subsets are often evaluated by the accuracy of the produced model. Exhaustive search is generally impractical, so at some defined stopping point, the subset of features with the highest score discovered up to that point is selected as the satisfactory feature subset. The stopping criterion varies with the selected feature selection algorithm. Possible criteria include: a subset score exceeding a threshold, the maximum allowed run time of an algorithm has been surpassed, etc.

In this work, support vector machines are used as a modeling tool to evaluate the subsets of features. Feature evaluation criteria is still an open discussion [22,24], as accuracy not always dictate a suitable performance score for a specific predictor, especially in medical applications, where in most cases the existing classes are unbalanced. The modeling technique and the used performance measures are described in the following. The search method is one of the focus issues in this paper and will be described in detail in Section 4.

2.2. SVM modeling

The sepsis problem is a very complex medical state, thus one of the main criteria for choosing a modeling technique is the capability of effectively representing highly nonlinear problems. Support vector machines is a method for learning separating functions into two-class classification problems [19]. The algorithm maps the nonlinear inputs to a high dimension feature space. In this feature space a linear classification surface is constructed. SVM are an easy to use classification technique even though users usually get unsatisfactory results at first due to poor parameter setup. The first version of SVM was introduced as a linear classifier. Given some training data $\mathcal{D}$, a set of n points of the form

$$\mathcal{D} = \{(\boldsymbol{u}_i, y_i) | \boldsymbol{u}_i \in \mathbb{R}^{N_t}, y_i \in \{-1, 1\}\}_{i=1}^{n} \tag{1}$$

where $\boldsymbol{u}_i$ are the input variables, y_i is either 1 or -1, indicating the class to which the input $\boldsymbol{u}_i$ belongs, and N_t is the total number of available input variables. Each $\boldsymbol{u}_i$ is a N_t-dimensional real vector. We want to find the maximum margin hyperplane that divides the points having $y_i = 1$ from those having $y_i = -1$. However, most classification problems can only be separated using a nonlinear classifier. The solution is to use a transformed *feature space* $\mathcal{F}$, mapping observations from a general set $\mathcal{D}$ into an inner product space $\mathcal{F}$:

$$\begin{aligned} &\phi : \mathbb{R}^{N_t} \to \mathcal{F}, \\ &\boldsymbol{u} \to \phi(\boldsymbol{u}), \end{aligned} \tag{2}$$

with ϕ being the nonlinear mapping function between $\mathbb{R}^{N_t}$ and $\mathcal{F}$. After this operation one works with the new database:

$$(\phi(\boldsymbol{u}_1), y_1), \ldots, (\phi(\boldsymbol{u}_n), y_n) \in \mathcal{F} \times Y. \tag{3}$$

The discriminant function will be in the form:

$$f(\boldsymbol{u}) = \left\langle \boldsymbol{w}, \boldsymbol{\phi}(\boldsymbol{u}) \right\rangle + \mathbf{b}, \tag{4}$$

where $\boldsymbol{w}$ are the normal vectors to the hyperplanes and $\mathbf{b}$ the respective offsets. In the feature space, $\mathcal{F}$, data is linearly separable. Further, when viewed in the original input space $\mathcal{D}$, f is a nonlinear function if $\phi(\boldsymbol{u})$ is a nonlinear function. Using the Lagrangian formulation of the problem, the training data will only appear in the form of dot products between vectors. This is a crucial property which allows the generalization of the linear procedure to the nonlinear case [25]. The Lagrange formulation of the dual problem in $\mathcal{F}$ becomes:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{maximize}} \quad & L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (\phi(\boldsymbol{u}_i) \cdot \phi(\boldsymbol{u}_j)) \\ \text{subject to :} \quad & 0 \le \alpha_i \le C, \quad i = 1, \ldots, n, \\ & \sum_{i=1}^{n} \alpha_i y_i = 0. \end{aligned} \tag{5}$$

where L_D is the objective function of the Lagrangian formulation of the dual problem, and α are the Lagrange multipliers. A *kernel function* is defined as:

$$k(\boldsymbol{u}_i, \boldsymbol{u}_j) = \phi(\boldsymbol{u}_i) \cdot \phi(\boldsymbol{u}_j). \tag{6}$$

With this function, it is only necessary to have the feature vector $\boldsymbol{u}_i$, since the scalar product $\phi(\boldsymbol{u}_i) \cdot \phi(\boldsymbol{u}_j)$ is directly calculated by computing the kernel $k(\boldsymbol{u}_i, \boldsymbol{u}_j)$.

Then, the optimization model in (5) can be solved using the method for solving the optimization in the separable case. This paper uses the *sequential minimal optimization* (SMO) [26] to solve the QP problem of SVM. The optimal hyperplane has the form:

$$\begin{aligned} f(\boldsymbol{u}, \alpha^*, b^*) &= \sum_{i=1}^{n} y_i \alpha_i^* \left\langle \phi(u_i), \phi(\boldsymbol{u}) \right\rangle + b^*, \\ &= \sum_{i=1}^{n} y_i \alpha_i^* k(\boldsymbol{u}_i, \boldsymbol{u}_j) + b^*. \end{aligned} \tag{7}$$

Depending upon the applied kernel, the bias b can be implicitly part of the kernel function. Therefore, if a bias term can be accommodated within the kernel function, the nonlinear SVM classifier is given by:

$$f(\boldsymbol{u}, \alpha^*, b^*) = \sum_{i=1}^{n} y_i \alpha_i^* \left\langle \phi(\boldsymbol{u}_i), \phi(\boldsymbol{u}) \right\rangle = \sum_{i=1}^{n} y_i \alpha_i^* k(\boldsymbol{u}_i, \boldsymbol{u}_j). \tag{8}$$

Some kernel functions include polynomial, radial basis function (RBF) and sigmoid kernel [25]. This paper uses the RBF kernel function (9) for the SVM classifier because this kernel function can analyze higher-dimensional data and only requires two parameters, C and γ [27].

$$k(\boldsymbol{u}_i, \boldsymbol{u}_j) = \exp(-\gamma \|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2). \tag{9}$$

In order to improve classification accuracy, these kernel parameters in the kernel functions should be properly set.

2.3. Objective function

In this work, feature selection is used with the purposes of improving modeling quality. The problem in study is a classification problem. The goals are the maximization of model performance and the minimization of the number of used features. The objective function is defined as a fitness function (as in EA), being our goal its maximization. The most suitable representation to the proposed task is [12]:

$$f(\boldsymbol{u}_i) = \alpha (1 - P) + (1 - \alpha) \left(1 - \frac{N_f}{N_t} \right), \tag{10}$$

where P is the classifier performance measure and N_f is the size of the tested feature subset. The term on the left side of the equation accounts for the overall accuracy and the term on the right for the percentage of used features. Note that both terms in the objective function are normalized. Constant $\alpha \in [0, 1]$ define the weights of the related goal; performance and subset size.

Traditionally, accuracy has been used to evaluate classifier performance. This measure is defined as the total number of good classifications over the total number of available examples. Usually most of the classification problems have two classes, positive and negative cases [28]. Thus, the classified test points can be divided into four categories that usually are represented in the well known confusion matrix: true positives (*TP*), true negatives (*TN*), false

positives (*FP*) and false negatives (*FN*). Given the four categories of the confusion matrix, accuracy is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}. \tag{11}$$

This criterion is limited, especially if one of the classes is much larger than the other. With a unbalanced classification problem, misclassifications in the minority class will not have a large impact in the accuracy value. Further, a good classification of a class might be more important than classifying other classes, and this cannot be assessed with accuracy. To account for these issues, additionally two other performance measures, commonly used in binary classification problems, were considered [29]:

$$\text{Sensitivity} = \frac{TP}{TP + FN}, \tag{12}$$

$$\text{Specificity} = \frac{TN}{FP + TN}. \tag{13}$$

Sensitivity is equal to the *TP* rate, i.e. the ratio of true positives that were identified in all the positive class samples. Analogously, specificity is the rate of *TN*. Sensitivity and specificity describe how well the classifier discriminates the positive and the negative classes, respectively.

The problem of finding an optimal feature subset using the objective function (fitness function) in (10) is highly complex. Thus, metaheuristics are used to search the space of feature combinations.

To solve this problem several search meta-heuristics have been proposed, as e.g. genetic algorithms (GA) [10], particle swarm (PSO) [9,27] and ant colony optimization (ACO) [12,30]. These methods are able to find fairly good solutions without searching the entire workspace. The feature selection technique proposed in this paper is a modified version of the binary PSO algorithm that optimize simultaneously the SVM model parameters.

3. Application of PSO in feature selection

Particle swarm optimization is a simple metaheuristic with biological inspiration in swarming behavior of some species [11]. Contrarily to evolutionary algorithms, it does not include any genetic operators, which makes it simpler than EA and also reduces the number of parameters to adjust. It has been applied to very different problems such as flow shop [31], antenna design [32] or healthcare [13,28].

The tendency to form swarms appears in many different organisms, for instance, in birds and fish. However, in a swarm there are no leaders, it is simply the result of the interaction between the behaviors of its individuals (local interactions) [33]. Swarming offers several advantages: protection against predators, more efficient reproduction (easier to find a mating partner), food search and gathering. Nevertheless, the search efficiency provided by swarming is what underlies particle swarm optimization (PSO) algorithms [11].

Initially, particle swarm optimization have been developed for continuous problems. Later, it was also extended for discrete problems, which resulted in the commonly known binary PSO [11]. The binary version of particle swarm optimization has the tendency to prematurely converge as noted in [13,31,34], especially in more challenging optimization tasks. There are various approaches to cope with this problem: use the mutation operator from EA [13], reset the swarm best if the fitness stagnates [34] or using perturbation mechanisms [31]. More recently, an hybrid PSO have been proposed in [9], but this algorithm has the disadvantage of being computationally heavy, which is not desired for a feature selection wrapper approach. A BPSO for feature selection and parameter determination of SVM is proposed in [27]. This paper proposes a modified binary PSO with SVM parameter determination.

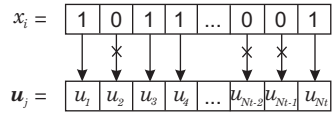

Fig. 1. Particle position decoding into selected features.

3.1. Formulation of the problem

In PSO, each particle (corresponding to an individual in EA) is a candidate solution of the optimization problem. A particle has a position and a velocity in the search space, where the method for updating the velocity depends on the particle itself and on the other particles. The first step when using a metaheuristic is to select the most suitable encoding scheme for the problem under study.

3.1.1. Encoding

The most common and generic encoding schemes are real and integer encoding. The use of each of them depends on the problem in hand. For the feature selection problem it is common to use a binary encoding. In this case, particles have their positions and velocities described by:

$$\mathbf{x}_j = (x_{1j}, x_{2j}, \ldots, x_{Nj}), \tag{14}$$

$$\mathbf{v}_j = (v_{1j}, v_{2j}, \ldots, v_{Nj}), \tag{15}$$

where x_{ij} and v_{ij} are binary variables, i is the particle index, and N is the total number of particles. Each particle has a position vector, in the binary space, with length equal to the number of data inputs (features). Thus, x_{ij} is binary, i.e. $x_{ij} \in \{0, 1\}$, with $j = 1, \ldots, N_t$, where N_t is the total (initial) number of features of the dataset. The variable x_{ij} corresponds to input u_j, where $j = 1, \ldots, N_t$. If feature u_j is to be selected then $x_{ij} = 1$, if not $x_{ij} = 0$, as depicted in Fig. 1.

3.1.2. Standard BPSO algorithm

The first step is to normalize the data, which is a crucial step in the classification process. By mapping the data between [0, 1] (or in another interval) we remove the scale effects, allowing underlying characteristics to be compared. Further, several parameters have to be selected, namely: number of particles N and number of iterations I. A common value for N is between 20 and 40. The positions of the first swarm can be initialized randomly by doing:

$$x_{ij} \leftarrow \begin{cases} 1, & \text{if } r \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \quad i = 1, \ldots, N, \quad j = 1, \ldots, N_t, \tag{16}$$

the velocities are given by:

$$v_{ij} = -v_{\max} + 2\,s\,v_{\max}, i = 1, \ldots, N, \quad j = 1, \ldots, N_t, \tag{17}$$

where r and s are random numbers $\in [0, 1]$, and $v_{\max}$ is the maximum value for the velocity. The above positions and velocities are iteratively updated based on the SVM classifier performance and the number of selected features. This process can be divided in to the following steps:

Step 1. **Evaluate each particle in the swarm**: in this step an SVM model is generated using the feature subset corresponding to the particle position. Then, the objective function defined in (10) is computed to evaluate the solutions.

Step 2. **Swarm and particle best values**: the particle best, $\mathbf{x}^{pb}$, corresponds to the position of the particle that had the best fitness f in all iterations. Swarm best, $\mathbf{x}^{sb}$, is the best position

achieved in all iterations by all the particles. This verification can be summarized by:

$$\mathbf{x}_i^{pb} \leftarrow \mathbf{x}_i, \quad \text{if } f(\mathbf{x}_i) > f(\mathbf{x}_i^{pb}) \tag{18}$$

$$\mathbf{x}^{sb} \leftarrow \mathbf{x}_i, \quad \text{if } f(\mathbf{x}_i) > f(\mathbf{x}^{sb}) \tag{19}$$

Step 3. **Update velocities**: velocity directs the movement in the search space taking into account the performance of the own particle and of the swarm, and it is updated as follows:

$$v_{ij} \leftarrow wv_{ij} + c_1 q\left(\frac{x_{ij}^{pb} - x_{ij}}{\Delta t}\right) + c_2 r\left(\frac{x_j^{sb} - x_{ij}}{\Delta t}\right), \quad i = 1, \ldots, N, j = 1, \ldots, N_t. \tag{20}$$

The term involving the constant c_1 is called the *cognitive component* and it measures the degree of self-confidence of a particle, the degree at which it trusts its performance. The term involving c_2 is the *social component* and it relies in the capability of the swarm to find better candidate solutions. The parameters q and r are uniform random numbers $\in[0, 1]$; Δt is the time step of each iteration; the term w is the inertia weight and it controls the influence of the previous velocity in the new velocity. If $w > 1$ the particle favors exploration over exploitation, else if $w < 1$ the particle gives more importance to the current best positions (particle best and swarm best). Thus, it is common to start with a value larger than $w = 1.4$ and then reduce it by a factor $\beta \in]0, 1[$ in each iteration. Further, after velocities have been update, the restriction $|v_{ij}| < v_{\max}$ is applied; this is a crucial step for the swarm to maintain coherence.

Step 4. **Update particle position**: the logistic function of the velocity is used as the probability distribution for the position [11,31]:

$$S(v_{ij}) = \frac{1}{1 + e^{-v_{ij}}} \tag{21}$$

Thus the particle position is calculated for each variable by:

$$x_{ij} \leftarrow \begin{cases} 0, & \text{if } r > S(v_{ij}) \\ 1, & \text{otherwise} \end{cases} \quad i = 1, \ldots, N, j = 1, \ldots, N_t \tag{22}$$

Step 5. **Continue the iterative process**: return to Step 1 if convergence or iteration limit is not achieved.

Unlike the continuous version of the PSO, in BPSO, particle position is restricted to the Hamming space. Thus, there is no risk of swarm divergence. However, the problem of premature convergence of the swarm arises. According to (21) and (22), if the velocity reaches a high value (e.g. $v_{ij} = 12$) it would become very difficult for new solutions to be tried. Hence, a velocity threshold is imposed, such that the probability of choosing a determined position is limited. Namely, the value $S(v_{\max})$ should be smaller than one, enabling some random "mistakes" to be made during position update. Note that in this situation the velocity clamping mechanism is similar to the mutation operator in the GA, with $v_{\max}$ playing the role of the mutation rate.

The value of $v_{\max}$ could, in principle, vary during search to initially favor exploration and later exploit the best solutions. However, to the best of our knowledge, such matter has never been addressed and will be regarded in the proposed modified binary PSO.

3.2. Convergence issues in BPSO

The binary version of particle swarm optimization (BPSO) has the tendency to prematurely converge, as noted in [35], especially in more challenging optimization tasks. There are various approaches to cope with this problem: using the mutation operator from EA [13], reset the swarm best if the fitness stagnates [34] or using perturbation mechanisms [31]. The modified BPSO algorithm proposed in this paper, associates the benefits of local search (mutations) [31,36] with resetting the swarm best mechanism [34].

3.3. Mechanisms to avoid premature convergence in PSO

These mechanisms have individual characteristics that contribute to an improvement of the solutions when there is premature convergence of the binary PSO.

3.3.1. Mutations

The importance of preserving the previous value of the velocity when calculating its current value has been discussed in Step 3. However, if a suboptimal solution happens to be better than any of the previously found, the swarm might converge towards that position. In such situation, the velocity of the particles will increase until $v_{\max}$. Therefore, it will be difficult to divert from this undesirable suboptimal position with a small change in velocity.

In order to explore untried areas of the search space, it was suggested in [36] to introduce small random mistakes in the current particle positions:

$$\begin{cases} x_{ij} = \neg x_{ij} & \text{if } r \le r_{mut}, \\ x_{ij} = x_{ij} & \text{otherwise,} \end{cases} \quad i = 1, \ldots, N, j = 1, \ldots, N_t. \tag{23}$$

where r_{mut} is the probability of random mutation. After updating the particle position as in (21) and (22), each of the bits of the position vector is mutated with a probability r_{mut}. Common values for the mutation probability are $r_{mut} = 1/N_t$, which means that at least one of the bits in the position vector will be flipped. The mutation rate should be sufficient to introduce some variability in the swarm without increasing too much the randomization of the algorithm.

This mechanism has proven to be effective in improving the solutions found by the original BPSO [36]. Nevertheless, in problems with higher dimensionality, increasing the value of r_{mut} might not be enough to avoid premature convergence.

3.3.2. Reset swarm best

This mechanism is used in the improved binary PSO (IBPSO) in [34]. Each particle adjusts its position according to two values, its own best solution so far, $\mathbf{x}^{pb}$, and the swarm best solution $\mathbf{x}^{sb}$. The particle best is a local search value, whereas the swarm best constitutes a global search value. If the $\mathbf{x}^{sb}$ value is itself trapped in a local optimum, this will limit the search of the entire swarm. Hence, when resetting $\mathbf{x}^{sb}$, the binary PSO will not be trapped in a local optimum, and better classification results can be achieved by searching for a new $\mathbf{x}^{sb}$ value in a region with a lower number of features. This process is depicted in Fig. 2. In this approach, all bits of $\mathbf{x}^{sb}$ are equal to 0 except one, which is randomly chosen and is set to 1. The algorithm is considered to have prematurely converged, see Fig. 2(a), when the value of $f(\mathbf{x}^{sb})$ stays constant for a chosen number of iterations I_{max}. The following step is to reset the swarm best, which consists of selecting any features. Thus, all the bits in the $\mathbf{x}^{sb}$ are changed to be zero. In most cases, the algorithm will converge towards a better solution, due to the reduction in the number of selected features while maintaining classification accuracy.

We combined the above mechanisms in a novel PSO (NPSO) approach, which was presented in [37]. The algorithm in [37] has demonstrated to have superior performance than the original BPSO and IBPSO when applied to feature selection. Nevertheless, NPSO had several disadvantages. One of them is the addition of two adjustable parameters, the random mutation probability r_{mut}, and

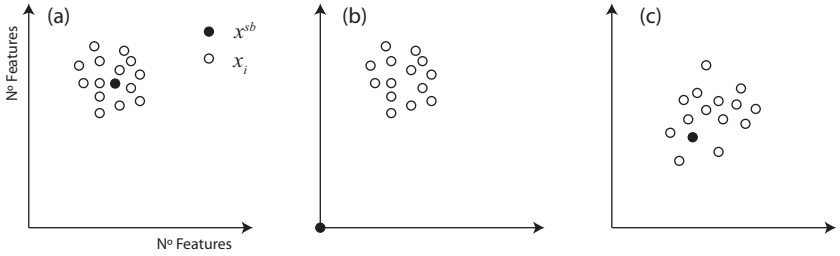

Fig. 2. Reset swarm best mechanism [34].

the maximum allowed number of iteration with a constant swarm best fitness $f(\boldsymbol{x}_{sb})$. Therefore, this paper proposes the modified BPSO algorithm, as described in the next section.

4. Modified BPSO

One of the major problems when using wrapper methods for feature selection is the process of selecting a proper set of parameters for the used model. This problem can be surpassed by automatically optimizing the SVM model parameters in parallel with the feature selection process [10]. The proposed modified binary PSO includes the SVM model parameters in the encoding of the particles and is described in the following.

4.1. Encoding

In this paper, the RBF kernel function, defined in (9), is used in the SVM classifier, once the RBF kernel function can analyze higher dimensional data and only requires two parameters, C and γ to be defined [38]. When the RBF kernel is selected, the parameters C and γ, and the features used as input attributes must be optimized using our proposed MBPSO system.

Therefore, the particle is in this case comprised of three parts, the features mask, C and γ. Fig. 3 shows the representation of particle i, where $j = 1, \ldots, N_P$ and N_P is the size of the particle. A part of the binary sequence refers to the feature selection (as described above) and the rest of the sequence to the model parameters. Nevertheless, the parameters are usually real valued, thus it is necessary to decode the binary strings in to floating point values. For each parameter of the model there is a specific part of the binary string that is decoded independently of a selected range.

4.2. Modified mechanisms to avoid premature convergence

The MBPSO combines the described *reset swarm best* mechanism with a *local search* operator that displaces the particle best position $\boldsymbol{x}^{pb}$ and uses an operator similar to the *mutation* mechanism, but instead of using a probability of mutation, the change in the particles are made controlling the value of $v_{\max}$. These modified mechanisms were first introduced in [39] and are used by the proposed MBPSO, once these mechanisms have shown better performance.

4.2.1. Local search

The $\boldsymbol{x}^{sb}$ is the only liaison between all the particles. Hence, the reset swarm best mechanism was kept from the NPSO, since it shown to be effective in preventing the premature convergence of $\boldsymbol{x}^{sb}$.

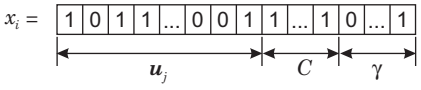

Fig. 3. Particle position decoding into selected features and SVM model parameters.

The criteria to reset the swarm best is to verify if the fitness of the $\boldsymbol{x}^{sb}$ stays constant for a given number of iterations. Usually, this value is around three iterations [34]. It is a small value to avoid the convergence of the rest of the swarm to a suboptimal $\boldsymbol{x}^{sb}$. The refinement of the best solution is difficult, and as a consequence NPSO takes a high number of iterations to converge to an acceptable solution. In [39], we have introduced a mechanism of local search that consists of displacing the $\boldsymbol{x}_i^{pb}$ values when resetting the swarm best:

$$\begin{cases} x_{ij}^{pb} = \neg x_{ij}^{pb} & \text{if } r \leq dr \\ x_{ij}^{pb} = x_{ij}^{pb} & \text{otherwise} \end{cases}, \quad i = 1, \ldots, N,\ j = 1, \ldots, N_P, \tag{24}$$

where dr is the displacement rate, i.e. the probability of each bit in the $\boldsymbol{x}_i^{pb}$ being flipped, and r is a random number $\in[0, 1]$. As can be seen in Fig. 4, even if most of the particles converge to a suboptimal $\boldsymbol{x}^{sb}$ (a), when this mechanism is applied the particles are displaced (b). The algorithm will be able to converge towards a better solution (c). Therefore, we will be able to refine the results around the best position reducing the risk of causing premature convergence of the algorithm.

4.2.2. Variability in the swarm

When the velocity reaches a high value (e.g. $|v_{ij}| = 10$) the probability of selecting a given bit is very close to one. The NPSO uses a mechanism similar to the mutations in EA to make small "mistakes" during position update, avoiding premature convergence. However, in the proposed MBPSO a more effective approach is applied, by controlling the value of $v_{\max}$.

It should be noticed that the probability of selecting a given bit when $v_{ij} = v_{\max}$ is:

$$\begin{cases} x_{ij} = 1, & p = S(v_{\max}) \\ x_{ij} = 0, & p = \overline{S}(v_{\max}) = 1 - S(v_{\max}) \end{cases} \tag{25}$$

and it can be proven that when $v_{ij} = -v_{\max}$ the probability of selecting each bit is:

$$\begin{cases} x_{ij} = 1, & p = S(-v_{\max}) = 1 - S(v_{\max}) \\ x_{ij} = 0, & p = \overline{S}(-v_{\max}) = 1 - S(-v_{\max}) = S(v_{\max}) \end{cases} \tag{26}$$

with $i = 1, \ldots, N$ and $j = 1, \ldots, N_P$. This means that the probability of making a "mistake" when the velocity achieves its maximum absolute value is equal to $1 - S(v_{\max}) = S(-v_{\max})$. Therefore, we can control the value of $v_{\max}$ in order to introduce variability in the swarm, instead of using the mutation operator as in NPSO. This approach has several advantages:

- localized variability – the variability is introduced where it is most important, when the particle position stagnates due to the saturation of the velocity;
- removing an extra operator – replacing the mutation operator with the control of $v_{\max}$ simplifies the algorithm. There will be less

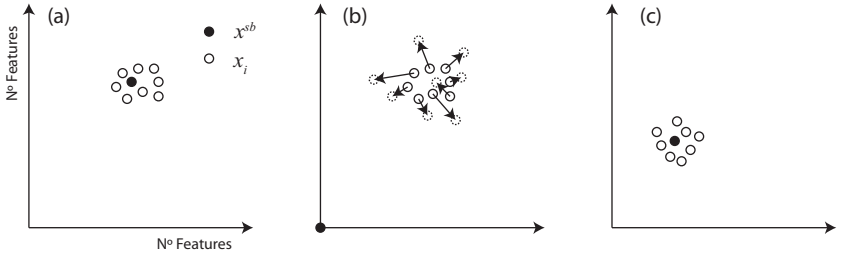

Fig. 4. Reset swarm best and local search.

Table 1
Databases used for validation of FS methods.

Number	Databases	Samples	Features	Classes
1	German (credit card)	1000	24	2
2	Sonar	208	60	2
3	WBCO	683 (699)	9	2
4	WPBC	198	32	2
5	WDBC	569	30	2
6	Colon cancer	62	2000	2

parameters to adjust and the algorithm will be computationally more efficient.

The proposed algorithm is summarized in Algorithm 1.

4.3. Benchmark tests

To demonstrate the improvements achieved by the proposed MBPSO approach, feature selection was performed in 6 benchmark datasets. The selected benchmark databases are listed in Table 1, which are available in the UCI repository [40]. This repository has been widely used by researchers as a primary source of machine learning databases (it has been cited over 1000 times). Further, the databases in this repository have a good balance between classes and a large diversity in feature number and sample size. Four methods are applied for searching the optimal subset of features:

1. The binary PSO (BPSO) using the mutation operator presented in [36];
2. The improved binary PSO (IBPSO) using the reset swarm best mechanism proposed in [34];
3. The genetic algorithm (GA) introduced in [10];
4. The proposed modified binary PSO (MBPSO) approach introduced in this paper.

Algorithm 1. Feature selection with the MBPSO

Initialize algorithm parameters
Normalize and divide data (train + test)
Randomly initialize swarm particles
while (number of iterations or stopping criteria are not met) **do**
  Evaluate the fitness of all the individuals
  for $i = 1$ to number of particles **do**
    if fitness of $\mathbf{x}_i$ is greater than the fitness of $\mathbf{x}_i^{pb}$ **then**
      $\mathbf{x}_i^{pb} = \mathbf{x}_i$
    end if
    if fitness of $\mathbf{x}_i$ is greater than the fitness of $\mathbf{x}^{sb}$ **then**
      $\mathbf{x}^{sb} = \mathbf{x}_i$
    end if
    if $\mathbf{x}^{sb}$ constant for I_{max} iterations **then**
      reset $\mathbf{x}^{sb}$
      update $\mathbf{x}^{pb}$ using displacement rate dr as in (24)
    end if
    for $j = 1$ to dimension of particle's position **do**
      $v_{ij} \leftarrow wv_{ij} + c_1 q \left(\frac{x_{ij}^{pb} - x_{ij}}{\Delta t} \right) + c_2 r \left(\frac{x_j^{sb} - x_{ij}}{\Delta t} \right)$
      if $|v_{ij}| > v_{max}$ **then**
        $v_{ij} = v_{max} \times sgn(v_{ij})$
      end if
      $S(v_{ij}) = \frac{1}{1+e^{-v_{ij}}}$
      if $(r < S(v_{ij}))$ **then**
        $x_{ij} = 1$ **else** $x_{ij} = 0$
      end if
    end for
    Introduce variability in the swarm using (25)
  end for
end while

Recall that SVM are used in combination with the optimization algorithms in a wrapper methodology, and the model parameters are optimized simultaneously in the feature selection optimization process. After FS, 10-fold cross validation is applied to the best subset in each algorithm and the best results are presented in Table 2.

In what concerns the comparison of the proposed MBPSO to other binary PSO algorithms, Table 2 shows clearly that the overall accuracy increases due to the proposed mechanisms to avoid premature convergence, while the number of selected features is smaller or similar using MBPSO.

Table 2 shows that the MBPSO introduces in general a slight increase in the specificity, and in some cases a significant improvement in the sensitivity. This is important since correct classifications for the positive cases are being made more accurately, and in many classification applications the positive class has a greater importance than the negative one. This is generally the case for medical applications, and for the case study presented in Section 5 is even more important, once the failure in correct classifications of the positive class means that a patient will have a larger risk of death.

Although the improvement of the results using MBPSO is modest, it is statistically significant as the p-value is <0.05. Comparing MBPSO with all the other binary PSO based algorithms, only for the WBCO database was not possible to improve the results in terms of accuracy.

The MBPSO approach is better than the IBPSO, due to the mechanisms that introduce variability in the swarm. Nevertheless, these differences are still dependant of the adjustment of the dr, since this parameter has to be well adjusted in order to compensate an increase in the reset $\boldsymbol{x}_{sb}$ iterations.

To further test the proposed MBPSO algorithm, the comparison with other metaheuristics used for feature selection is very important. The application of GA to feature selection is well studied in the literature, so it presents a solid ground of comparison. From Table 2 it is possible to observe that the MBPSO results in terms of accuracy are similar or slightly better than GA. However, MBPSO can find smaller feature subsets. In terms of sensitivity, GA is better, except for the Colon database. On the other hand, GA has a lower specificity.

Table 2
Comparison of optimization algorithms for feature selection using six benchmark datasets.

Database	Method	Accuracy		Nr. features		Sensitivity		Specificity	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
German	No-FS	75.90	3.41	24	0	40.00	6.67	91.29	3.40
	BPSO	72.07	2.56	4	1	37.65	7.65	91.34	3.49
	IBPSO	74.61	2.22	3	1	38.43	8.37	90.87	2.95
	GA	76.70	2.41	4	1	46.67	8.75	89.57	2.86
	MBPSO	76.00	2.62	3	1	37.00	9.36	92.71	3.33
Sonar	No-FS	89.47	3.63	60	0	93.64	6.14	84.78	9.87
	BPSO	88.86	4.98	13	2	90.45	5.97	86.76	8.27
	IBPSO	91.79	5.45	12	2	94.38	5.37	87.48	9.04
	GA	93.76	5.99	14	3	97.35	4.27	89.78	13.54
	MBPSO	93.74	5.07	12	2	93.64	6.14	93.78	7.15
WBCO	No-FS	96.92	2.03	9	0	96.61	4.43	97.07	2.14
	BPSO	96.31	2.37	2	1	96.27	3.54	97.16	2.12
	IBPSO	96.54	2.17	2	1	95.34	3.93	96.84	2.65
	GA	96.49	1.98	2	1	97.92	2.20	95.72	2.25
	MBPSO	96.34	2.79	2	1	95.82	3.40	96.61	2.88
WPBC	No-FS	77.35	2.15	32	0	4.00	8.43	100.00	0.00
	BPSO	78.53	5.45	3	1	31.75	21.34	91.38	5.23
	IBPSO	78.04	7.04	3	1	29.95	19.34	89.65	5.34
	GA	77.98	6.30	4	1	33.50	20.42	91.95	5.25
	MBPSO	78.90	8.60	3	1	28.00	27.30	94.62	5.27
WDBC	No-FS	95.96	3.00	30	0	93.87	7.10	97.21	2.62
	BPSO	96.85	1.32	3	1	94.34	4.43	97.85	2.37
	IBPSO	97.34	1.48	3	1	93.12	4.04	98.47	1.83
	GA	98.06	1.56	3	1	96.69	3.20	98.88	1.95
	MBPSO	97.71	1.20	3	0	95.74	4.17	98.88	1.44
Colon	No-FS	89.05	12.86	2000	0	81.67	24.15	92.50	12.08
	BPSO	94.26	7.34	16	12	94.32	8.34	93.46	8.38
	IBPSO	95.84	8.83	18	10	95.43	7.83	96.34	6.73
	GA	97.14	9.04	22	28	96.67	10.54	97.50	7.91
	MBPSO	98.33	5.27	14	14	100.00	0.00	97.50	7.91

5. Application to sepsis outcome prediction

Sepsis is the 10th leading cause of death in the United States, being one of the only two infectious diseases listed in the top 15 causes of death [41]. According to a study conducted by Emory University School of Medicine and the Centers for Disease Control and Prevention, the rate of patients with sepsis in the US population has increased more than 300 percent between 1979 and 2000 [42]. Furthermore, given that one of the most important risk factors of sepsis is the advanced age, the disease rates will very likely continue to grow with the increasing ageing of the population [43].

This disease is not only responsible for a high number of casualties, but it also represents a large financial burden to the health care system. For example in USA, a sepsis patient stays on average 19.6 days in an intensive care unit (ICU), with an average cost of $22,100. This reflects in a total of approximately $1.7 billion per year for treating patients with severe sepsis [43]. An estimated 20–30 percent of these costs are related to the treatment of the illness, while 70–80 percent are associated with productivity losses due to mortality [15].

It has been proven that an *early goal directed therapy* (EGDT) is more effective in reducing mortality risk of patients with sepsis than standard care [44]. However, it involves the identification of high-risk patients, namely the prediction of the transition between sepsis and more severe states of sepsis. For the case of neonates, it was possible to identify some biomarkers that enable the detection and early diagnosis of sepsis [45]. However, in the case of adult patients this identification was not yet possible, which drastically reduces the applicability of EGDT.

The Acute Physiology and Chronic Health Evaluation (APACHE II) is a classification system for disease severity, designed to be used in the evaluation of adult patients. The Simplified Acute Physiology Score (SAPS II) was developed with the primary goal of simplifying the APACHE II score [46]. The Sequential Organ Failure Assessment (SOFA) score is a scoring system, which evaluates the extent organ function or rate of failure. Finally, Multiple Organ Dysfunction Score (MODS) has similar purposes to the SOFA score, but is computed in a different manner.

More recently, there has also been some work developed to determine disease severity using machine learning algorithms. This approach consists of using knowledge discovery techniques in order to predict the health condition of sepsis patients. Most of the developed tools run on static data (e.g. age, gender, blood pressure, hearth rate, temperature, pH), and are used to classify if it is probable that a patient will evolve to a more severe health state. The number of developed methods is scarce, the most known approaches are [46–49]. In [46,47], an application of a knowledge-based neural network technique is presented, and uses 12 of most frequently measured variables (in patients with sepsis) to predict the outcome of a sepsis patient (decease/survive). In [48], an extension of [46,47] is presented, using 28 variables instead of the initial 12, achieving considerably better results. Finally, in [49], an early warning system (EWS) is proposed, where a multivariate logistic regression model is used to provide prior warnings for septic shock.

The methodology used in this work, in what concerns the application of feature selection methods to the problem of sepsis outcome prediction, is a follow up of the work developed in [48], and aims not only to derive an accurate classifier, but also shade some light in which physical variables are more likely to dictate the patient state of health under a sepsis clinical state.

In such a case the most important issue is to have a classifier with the best accuracy possible. The models generated by SVM will be optimized through feature selection using the proposed MBPSO

Table 3
Sepsis databases.

Features	Number of patients	Samples	Classes	Reference
12	89	2878	2	[47]
28	121	2055	2	[48]

algorithm. Thus, in this section this method will be validated against GA, in order to prove its multiple advantages.

This paper uses the public available MEDAN database [46], containing data of 382 patients with abdominal septic shock, recorded from 71 German intensive care units (ICUs), from 1998 to 2002. This database holds: personal records, physiological parameters, procedures, diagnosis/therapies, medications, and respective outcomes (survived or deceased). For the purpose of the present work, we will focus exclusively in physiological parameters, which include a total number of 103 different variables/features, which is the same database used in [47,48].

5.1. Data preprocessing

We follow the preprocessing procedures done in [48]. From the initial set containing a total of 103 features, two different subsets of features were chosen as inputs for our models. One, was defined as in [47], for comparison purposes. It contains the 12 most frequently measured variables: creatinin (mg/dl), calcium (mmol/l), arterial, pCO_2 (mmHg), pH, haematocrit (%), sodium (mmol/l), leukocytes (1000/ml), haemoglobin (g/dl), central venous pressure (CVP) (cmH_2O), temperature (C), heart frequency (1/min) and systolic blood pressure (mmHg). A total of 121 patients were found to have data regarding these features. However, the previous subset of features was considered to be too narrow, and in [48] a second data subset was defined including a total of 28 variables, which were found to be present in a total of 89 patients. These two sets of features are described in Table 3.

All the optimization algorithms presented in this work were applied to each of these subsets, using SVM as learning machines. The reason to apply feature selection techniques to these not so large subsets is related to the clinical relevance of finding the specific variables that relate the most with the prediction of a patient's outcome. The accuracy, or the correct classification rate, is not always the best way to evaluate the performance of the classifier. For this particular application, the main interest is to correctly classify which patients are more likely to die, in order to rapidly act in their best interest. Bearing this in mind the classifier should classify as accurate as possible the death cases, or the true positives, and have a low number of false negatives. In other words, the classifier should maximize sensitivity. In order to evaluate the performance of the developed prediction models, upon the described subsets of data, three different criteria were used: accuracy, sensitivity and specificity.

5.2. Results using 12 features

The benefits of feature selection over the 12 feature dataset are summarized in Table 4. Comparing the results in terms of accuracy before and after feature selection, a slight increase can be observed.

Table 4
Comparison of FS techniques over the 12 features sepsis database.

Method	Accuracy	NF	Sensitivity	Specificity
No-FS	72.6 ± 2.6	12	47.6 ± 5.6	88.1 ± 2.8
BPSO	73.4 ± 2.1	2 ± 1	54.3 ± 3.9	87.4 ± 3.1
IBPSO	75.3 ± 3.0	3 ± 1	55.3 ± 5.2	88.6 ± 2.3
GA	75.0 ± 3.8	3 ± 1	52.2 ± 6.2	89.1 ± 3.5
MBPSO	76.5 ± 1.8	2 ± 1	60.4 ± 4.1	86.5 ± 1.9

Table 5
Comparison of FS techniques over the 28 features sepsis database.

Method	Accuracy	NF	Sensitivity	Specificity
No-FS	89.0 ± 1.7	28	76.1 ± 5.4	95.6 ± 1.7
BPSO	94.0 ± 1.5	6 ± 1	89.5 ± 5.6	96.1 ± 2.0
IBPSO	94.2 ± 1.1	6 ± 1	90.4 ± 5.3	95.9 ± 1.8
GA	95.7 ± 1.4	7 ± 1	94.3 ± 1.2	96.5 ± 2.1
MBPSO	94.4 ± 1.2	6 ± 1	90.2 ± 5.1	96.5 ± 1.9

However, the value of sensitivity increased considerably after feature selection, without a noticeable decrease in specificity. This is an increase in the quality of the model, once the sensitivity is very important for this application. Further, a small increase in the overall accuracy was registered. This is due to the reduction in the feature subset size from 12 (initial) to 3 (after FS) features on average, which implies a much less complex model. The best overall accuracy was achieved by MBPSO, with a significant improvement in terms of sensitivity, and using only two features.

5.3. Results using 28 features

The results for the 28 variables dataset are depicted in Table 5. The results show a considerable increase in the predictive performance of the generated classifier.

Regarding Table 5, it can be seen that the accuracy over the 28 features dataset improved after feature selection. Also, the initial sensitivity was improved, while maintaining a good specificity. This can probably be justified by the reduction of the number of redundant and noisy feature used to generate the SVM surface. In terms of the number of selected features, only 20% of the initial features are used in the model with the best performance.

The accuracy is substantially better when using 28 features, meaning that some meaningful features were left out in the first 12 features dataset. However, the average number of selected features more than doubles when using 28 features instead of 12. Regarding the performance of the used optimization algorithms, in the improvement of sepsis outcome prediction, both GA-SVM and MBPSO-SVM have considerably better accuracies than the remaining algorithms, including the initial set of features. These differences are statistically relevant with more than 95% of confidence, since the calculated p-values, with respect to the accuracy without feature selection (No-FS), are much smaller than 0.05. There is no significant difference in performance between GA-SVM and MBPSO-SVM. The set of initial features is in both cases reduced, on average, to 1/4 of the initial size. However, the MBPSO running time was on average approximately half of GA run time (for example for the 12 feature data set the running time was 8000 s for the GA vs 4500 s for the MBPSO).

5.4. Comparison with previous work

A comparison was made between the studied wrapper methods and other FS methods already applied to the problem of sepsis outcome prediction. These wrapper approaches are: neuro-fuzzy modeling using only the 12 feature dataset [47], bottom-up (BU) using fuzzy modeling and neural networks [48] and ant feature selection (AFS) using fuzzy modeling and neural networks [48]. The results are depicted in Tables 6 and 7 for the 12 and 28 features datasets respectively.

The differences between the proposed approaches and the other methods are significant. The GA-SVM and MBPSO-SVM have superior performances in all the measures that evaluate the generalization capabilities of the generated model (e.g. accuracy, sensitivity and specificity). The best performance of all wrapper

Table 6
Comparison with previous work for the 12 features MEDAN dataset.

Method	Accuracy	Nr. features	Sensitivity	Specificity
Neuro-fuzzy [47]	69.0 ± 4.4	12	15.01 ± –	92.3 ± –
BU-fuzzy [48]	74.1 ± 1.3	4 ± 2	79.9 ± 2.6	71.2 ± 2.9
AFS-fuzzy [48]	72.8 ± 1.4	2 ± 1	76.5 ± 0.0	70.5 ± 0.0
BU-NN [48]	73.2 ± 2.0	5 ± 3	54.5 ± 5.4	81.7 ± 3.6
AFS-NN [48]	75.7 ± 1.4	5 ± 3	59.6 ± 0.0	85.6 ± 0.0
No-FS SVM	72.6 ± 2.6	12	47.6 ± 5.6	88.1 ± 2.8
GA-SVM	75.0 ± 3.8	3 ± 1	52.2 ± 6.2	89.1 ± 3.5
MBPSO-SVM	76.5 ± 1.8	2 ± 1	60.4 ± 4.1	86.5 ± 1.9

Table 7
Comparison with previous work for the 28 features MEDAN dataset.

Method	Accuracy	Nr. features	Sensitivity	Specificity
Neuro-fuzzy[47]	–	–	–	–
BU-fuzzy [48]	82.3 ± 1.6	5 ± 2	82.3 ± 1.6	83.3 ± 2.6
AFS-fuzzy [48]	78.6 ± 1.4	6 ± 3	79.2 ± 0.0	78.2 ± 0.0
BU-NN [48]	81.2 ± 2.0	6 ± 2	64.2 ± 3.9	90.3 ± 2.1
AFS-NN [48]	81.9 ± 2.2	8 ± 4	67.0 ± 0.1	90.2 ± 0.0
No-FS SVM	89.0 ± 1.7	28	76.1 ± 5.4	95.6 ± 1.7
GA-SVM	95.7 ± 1.4	7 ± 1	94.3 ± 1.2	96.5 ± 2.1
MBPSO-SVM	94.4 ± 1.2	6 ± 1	90.2 ± 5.1	96.5 ± 1.9

methods is achieved over the 28 feature database by the GA-SVM, but with similar performance to MBPSO-SVM.

This comparison is solely illustrative, since the methods use different modeling techniques. However, this might be an indicator that SVM have better generalization properties over this database than fuzzy modeling or neural networks.

6. Conclusions

The proposed MBPSO algorithm was tested over 6 benchmark databases, showing a better performance than the state of the art methods for PSO and similar or better results than GA. Some limitations to the proposed MBPSO algorithm can arise if there is a poor adjustment in the parameters, as it has two more adjustable parameters than the standard version of binary PSO. Nevertheless, it has only one more parameter than genetic algorithm which has a similar overall performance. Despite all the mechanisms for premature convergence, the proposed MBPSO algorithm was able to correctly estimate the parameters of support vector machines, in parallel to feature selection. The generation of models for the mortality risk evaluation in patients with sepsis is a very important topic in medicine. The application of SVM based feature selection to the case study of sepsis outcome prediction shown to be superior than all the previously used methods [47,48,50].

Future work considers experimenting the introduced algorithm with other medical databases in order to more consistently compare its performance with other feature selection techniques.

Acknowledgment

This work is supported by the Portuguese Government under the programs: Strategic Project, PEst-OE/EME/LA0022/2011, through FCT (under the Unit IDMEC – Pole IST, Research Group IDMEC/LAETA/CSI), project PTDC/SEM-ENR/100063/2008, and by a FCT post-doctoral grant SFRH/BPD/65215/2009, Fundação para a Ciência e a Tecnologia (FCT), Ministério da Educa cão e Ciência, Portugal.

References

[1] D.D. Meisel, Fourier transforms of data sampled in unequally spaced segments, Astronomical Journal 84 (1979) 116–126.
[2] P. Bajcsy, An overview of DNA microarray grid alignment and foreground separation approaches, EURASIP Journal on Applied Signal Processing 2006 (2006) 1–13.
[3] E. Andreou, E. Ghysels, A. Kourtellos, Oxford Handbook on Economic Forecasting – Forecasting with Mixed-frequency Data, Oxford University Press, USA, Oxford, 2010.
[4] W.H. Press, G. Rybicki, Annotation: what can be done about missing data? Astrophysical Journal 338 (1989) 277–280.
[5] N.I. Kalyadin, V.A. Lemenkov, I.R. Losev, S.I. Kantor, Problems of medical monitoring of patients and the requirements for development of computer monitoring systems, Biomedical Engineering 30 (2) (1996) 81–85.
[6] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From data mining to knowledge discovery in databases, AI Magazine 17 (1996) 37–54.
[7] S.N. Ghazavi, T.W. Liao, Medical data mining by fuzzy modeling with selected features, Artificial Intelligence in Medicine 43 (2008) 195–206.
[8] S.M. Vieira, J. ao, M.C. Sousa, U. Kaymak, Fuzzy criteria for feature selection, Fuzzy Sets and Systems 189 (1) (2012) 1–18.
[9] C.-L. Huang, J.-F. Dun, A distributed PSO-SVM hybrid system with feature selection and parameter optimization, Applied Soft Computing 8 (4) (2008) 1381–1391.
[10] C.-L. Huang, C.-J. Wang, A GA-based feature selection and parameters optimization for support vector machines, Expert Systems with Applications 31 (2) (2006) 231–240.
[11] M. Wahde, Biologically Inspired Optimization Methods, 1st edition, WIT Press, Southampton, UK, 2008.
[12] S.M. Vieira, J.M.C. Sousa, T.A. Runkler, Two cooperative ant colonies for feature selection using fuzzy models, Expert Systems with Applications 37 (2010) 2714–2723.
[13] E. Alba, J. Garcia-Nieto, L. Jourdan, E.-G. Talbi, Gene selection in cancer classification using PSO/SVM and GA/SVM hybrid algorithms, in: Proc. of the IEEE Congress on Evolutionary Computation, 2007, pp. 284–290.
[14] R.C. Bone, R.A. Balk, F.B. Cerra, R.P. Dellinger, A.M. Fein, W.A. Knaus, R.M. Schein, W.J. Sibbald, Definitions for sepsis and organ failure and guidelines for the use of innovative therapies in sepsis. The ACCP/SCCM Consensus Conference Committee. American College of Chest Physicians/Society of Critical Care Medicine., Chest 101 (6) (1992) 1644–1655.
[15] H. Burchardi, H. Schneider, Economic aspects of severe sepsis: a review of intensive care unit costs, cost of illness and cost effectiveness of therapy, PharmacoEconomics 22 (12) (2004) 793–813.
[16] A. Fialho, F. Cismondi, S. Vieira, S. Reti, J. Sousa, S. Finkelstein, Data mining using clinical physiology at discharge to predict ICU readmissions, Expert Systems with Applications 39 (18) (2012) 13158–13165.
[17] M.M. Levy, M.P. Fink, J.C. Marshall, E. Abraham, D. Angus, D. Cook, J. Cohen, S.M. Opal, J.-L. Vincent, G. Ramsay, 2001 SCCM/ESICM/ACCP/ATS/SIS International Sepsis Definitions Conference, Intensive Care Medicine 29 (2003) 530–538.
[18] R. Kohavi, G. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1–2) (1997) 273–324.
[19] V. Kecman, Learning and Soft Computing Support Vector Machines, Neural Networks, Fuzzy Logic Systems, MIT Press, Massachusetts, USA, 2001.
[20] B. Hammer, K. Gersmann, A note on the universal approximation capability of support vector machines, Neural Processing Letters 17 (2003) 43–53.
[21] A. Ben-Hur, C.S. Ong, S. Sonnenburg, B. Schölkopf, G. Rätsch, Support vector machines and kernels for computational biology, PLoS Computational Biology 4 (10) (2008).
[22] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh (Eds.), Feature Extraction: Foundations and Applications, Springer, New York, USA, 2006.
[23] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.
[24] S.M. Vieira, U. Kaymak, J.M.C. Sousa, Cohen's kappa coefficient as a performance measure for feature selection., in: Proc. of 2010 IEEE World Congress on Computational Intelligence, WCCI 2010, IEEE Int. Conf. on Fuzzy Sys., Barcelona, Spain, 2010, pp. 1–8.
[25] C.J. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (1998) 121–167.
[26] J.C. Platt, Sequential minimal optimization: a fast algorithm for training support vector machines, Tech. rep., Microsoft Research, 1998.
[27] S.-W. Lin, K.-C. Ying, S.-C. Chen, Z.-J. Lee, Particle swarm optimization for parameter determination and feature selection of support vector machines, Expert Systems with Applications 35 (4) (2008) 1817–1824.
[28] M.A. Mazurowski, P.A. Habas, J.M. Zurada, J.Y. Lo, J.A. Baker, G.D. Tourassi, Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance, Neural Networks 21 (2–3) (2008) 427–436.
[29] F. Cismondi, A.L. Horn, A.S. Fialho, S.M. Vieira, S.R. Reti, J.M.C. Sousa, S. Finkelstein, Multi-stage modeling using fuzzy multi-criteria feature selection to improve survival prediction of ICU septic shock patients, Expert Systems with Applications 39 (16) (2012) 12332–12339.
[30] C.-L. Huang, ACO-based hybrid classification system with feature subset selection and model parameters optimization, Neurocomputing 73 (1–3) (2009) 438–448.
[31] L. Yuan, Z.-D. Zhao, A modified binary particle swarm optimization algorithm for permutation flow shop problem, in: Proc. of the International Conference on Machine Learning and Cybernetics, vol. 2, 2007, pp. 902–907.
[32] A. Marandi, F. Afshinmanesh, M. Shahabadi, F. Bahrami, Boolean particle swarm optimization and its application to the design of a dual-band dual-polarized

planar antenna, in: Proc. of IEEE Congress on Evolutionary Computation, 2006, pp. 3212–3218.
[33] C.W. Reynolds, Flocks, herds and schools: a distributed behavioral model, SIGGRAPH Computer Graphics 21 (1987) 25–34.
[34] L.-Y. Chuang, H.-W. Chang, C.-J. Tu, C.-H. Yang, Improved binary PSO for feature selection using gene expression data, Computational Biology and Chemistry 32 (1) (2008) 29–38.
[35] Z.-H. Zhan, J. Zhang, Y. Li, H.-H. Chung, Adaptive particle swarm optimization, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 39 (6) (2009) 1362–1381.
[36] S. Lee, S. Soak, S. Oh, W. Pedrycz, M. Jeon, Modified binary particle swarm optimization, Progress in Natural Science 18 (9) (2008) 1161–1166.
[37] G.J. Farinha, S.M. Vieira, L.F. Mendonça, J.M. Sousa, Optimization of fuzzy models using a novel PSO algorithm: application to medical databases, in: Proc. of the 18th World Congress of the International Federation of Automatic control, 2011, p. 6.
[38] C.W. Hsu, C.C. Chang, C.J. Lin, A practical guide to support vector classification, Tech. rep., National Taiwan University, Taipei, 2003.
[39] S.M. Vieira, L.F. Mendonça, G.J. Farinha, J.M. Sousa, Metaheuristics for feature selection: application to sepsis outcome prediction., in: Proc. of the World Congress on Computational Intelligence 2012, WCCI-CEC2012, 2012, pp. 2395–2402.
[40] A. Asuncion, D. Newman, UCI machine learning repository, 2007. URL www.archive.ics.uci.edu/ml
[41] H.-C. Kung, D.L. Hoyert, J. Xu, S.L. Murphy, Deaths: final data for 2005, National Vital Statistics Reports 56 (10) (2008) 1–120.
[42] G.S. Martin, D.M. Mannino, S. Eaton, M. Moss, The epidemiology of sepsis in the United States from 1979 through 2000, The New England Journal of Medicine 348 (16) (2003) 1546–1554.
[43] D. Angus, W.T. Linde-Zwirble, J. Lidicker, G. Clermont, J. Carcillo, M.R. Pinsky, Epidemiology of severe sepsis in the United States: analysis of incidence, outcome, and associated costs of care, Critical Care Medicine Division 29 (7) (2001) 1303–1310.
[44] R.M. Otero, H.B. Nguyen, D.T. Huang, D.F. Gaieski, M. Goyal, K.J. Gunnerson, S. Trzeciak, R. Sherwin, C.V. Holthaus, T. Osborn, E.P. Rivers, Early goal-directed therapy in severe sepsis and septic shock revisited*, Chest 130 (5) (2006) 1579–1595.
[45] J.R. Moorman, D.E. Lake, M.P. Griffin, Heart rate characteristics monitoring for neonatal sepsis, IEEE Transactions on Biomedical Engineering 53 (1) (2006) 126–132.
[46] E. Hanisch, R. Brause, J. Paetz, B. Arlt, Review of a large clinical series: predicting death for patients with abdominal septic shock, Journal of Intensive Care Medicine 26 (1) (2011) 27–33.
[47] J. Paetz, Knowledge-based approach to septic shock patient data using a neural network with trapezoidal activation functions, Artificial Intelligence in Medicine 28 (2) (2003) 207–230.
[48] A.S. Fialho, F. Cismondi, S.M. Vieira, J.M. da Costa Sousa, S.R. Reti, M.D. Howell, S.N. Finkelstein, Predicting outcomes of septic shock patients using feature selection based on soft computing techniques, in: E. Hüllermeier, R. Kruse, F. Hoffmann (Eds.), IPMU(2), vol. 81 of Communications in Computer and Information Science, Springer, New York, USA, 2010, pp. 65–74.
[49] D. Shavdia, Septic shock: providing early warnings through multivariate logistic regression models, Master's thesis, Massachusetts Institute of Technology, September 2007.
[50] J. Paetz, B. Arlt, K. Erz, K. Holzer, R. Brause, E. Hanisch, Data quality aspects of a database for abdominal septic shock patients, Computer Methods and Programs in Biomedicine 75 (1) (2004) 23–30.