Biologia Serbica, 2017, 39(1):83-98

DOI 10.5281/zenodo.827099

Minirewiev

Optimization, classification and dimensionality reduction in biomedicine and bioinformatics

Vladimir Filipović*

Department of Computer Science, Faculty of Mathematics, University of Belgrade, Studentski trg 16, 11000 Belgrade, Serbia

Summary: Research in biomedicine is faced with various problems connected to high-throughput processing – the need to handle the high frequency of incoming data and its high-dimensionality by means of a large number of measured features. Biomedicine needs efficient methods to deal with the enormous amount of collected data as well as effective tools to extract meta-data and information. It needs methods to explore data by means of classification and to evaluate data and models with respect to accuracy and reliability. Optimization methods have been successfully applied to these problems, but the complexity of the data, i.e. varying data density, high dimensionality and model reliability, is still very challenging. This paper addresses some important issues concerning the classification of a large amount of data: k-nearest-neighbor (kNN)-based and support vector machine (SVM)-based classification, dimensionality reduction for kNN and SVM classification, and optimal parameter settings for a SVM-based classifier. Dimensionality reduction and parameter selection are accomplished by using an electromagnetism-like metaheuristic (EM). The same EM is used for solving another optimization problem studied in this paper – the maximum betweenness problem (MBP). During radiation hybrid experiments, X-rays are used to fragment the chromosome. The probability that the given dose of an X-ray will break the chromosome rises with the distance between chromosomes. In this way, markers are placed on two separate chromosomal fragments. By estimating the frequency of the breaking points, and thus the distances between markers, it is possible to determine their order in a manner analogous to meiotic mapping. In this context, improvement of the radiation experiment is achieved by solving the MBP, i.e. by determining the total ordering of the markers that maximizes the number of satisfied constraints.

Keywords: classification; electromagnetism-like metaheuristic; optimization; support vector machine.

Introduction

In recent years, there has been a tremendous growth of interest in applications of optimization in biological and medical sciences. In many areas of biomedicine, optimization has become an indispensable tool (Pardalos and Romeijn 2009). It is used in various contexts, e.g. in diagnostic and prognostic systems for medical data analysis (Hammer and Bonates 2006), for hemodialysis schedule optimization (Choi et al. 2017), in protein fold recognition (Yan et al. 2017), etc. Optimization is also frequently used for designing and modeling complex systems, which are essential in biomedical and biological research, e.g. in solving the maximum betweenness problem (Filipović et al. 2013), the highly connected deletion problem (Hüffner et al. 2014), etc.

Exhaustive enumeration of all optimization techniques applied to biomedicine is far beyond the scope of this paper and can be investigated in the works of Pardalos et al. (2005) and Pardalos and Romeijn (2009), which give an excellent review of research in this field.

Data mining and classification

Data mining is one of the most popular and exciting disciplines of applied informatics. It allows researchers to discover complex and hidden patterns in data, which can potentially lead to completely new conclusions in different disciplines, where sometimes even experts in the disciplines cannot do better. Nowadays, there is an extremely rapid growth in the volume of data stored in biological databases,

e-mail: vladaf@matf.bg.ac.rs

with increased complexity of data and a very high dimensionality. One particularly active area in biomedicine and bioinformatics is the development and application of data mining algorithms to obtain more useful information from large sets of semi-structured or even unstructured biological data. Readers interested in this theme are encouraged to study review (Lavrač 1999; Kononenko 2001; Pardalos et al. 2007) and position papers (Patel et al. 2009; Seffert et al. 2011).

Data mining includes **classification**, which predicts a certain outcome based on a given input. An illustrative example of a classification task is given in Fig. 1. In order to learn how to predict outcome, the algorithm uses a set of training records containing a set of attributes and the respective outcome (Pardalos et al. 2007). The classification algorithm then, in the so-called training phase, tries to discover relationships between the attributes that would make it possible to predict an outcome. After this step, the algorithm is given a dataset not seen before, called a set of testing records, which contains the same set of attributes, except for the prediction attribute that is not yet known. The algorithm analyses the input and produces a prediction – this is the testing phase. The prediction accuracy defines the quality of the classification algorithm. After the testing phase, classifier is used in real-life conditions. The classification process is described by the flowchart in Fig. 2.

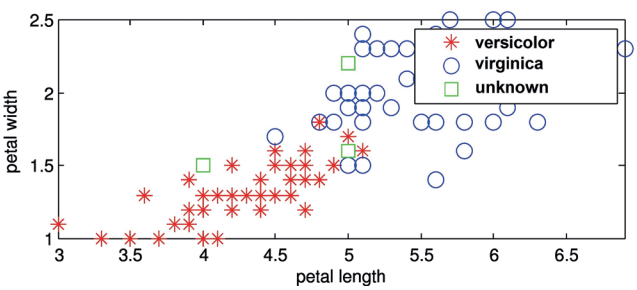

Fig. 1. Basic concepts of classification.

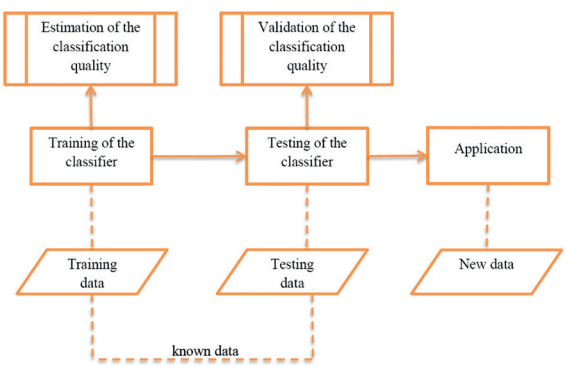

Fig. 2. Classification process.

The illustrative example of a classification task provided in Fig. 1, with only two possible outcomes (in versicolor and virginica), is also an example of a binary classification problem. In the **binary** classification problem, the training dataset is composed of feature vectors labelled with one of the two possible classes. **Multiclass** classification problems, where feature vectors are labeled with more than two classes, as shown in Fig. 3, can be reduced to multiple binary classification problems (Allwein et al. 2001).

Classification can be based on mathematical models, heuristic models or random models. During the past decades, many powerful and robust classification methods have been developed. This paper will deal with two of them: the k-nearest neighbor (kNN) and the support vector machine.

1) The k-nearest neighbor (k-NN) classifier is a non-parametric classification technique that classifies objects based on the set of closest training examples in the feature space (Pardalos et al. 2007; Grbić et al. 2016). More precisely, let us consider the binary classification problem where D_{tr} is a training set composed of N_{tr} pairs $(\boldsymbol{x}_i, y_i)$, $i=1,\ldots,N_{tr}$, where $\boldsymbol{x}_i \in R^N$ is a N-dimensional feature vector and $y_i \in \{-1,1\}$ is a corresponding class label. Given a new training record x_{new} for which the class should be predicted, k-NN finds a set of k training records $\{\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2},\ldots,\boldsymbol{x}_{ik}\}$ that are closest to x_{new} with regard to the predefined distance function. A classifier predicts that the class of x_{new} is a more frequent class among the training records $\{\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2},\ldots, \boldsymbol{x}_{ik}\}$. The result of work of a 3-NN classifier on one record (marked with a triangle) is shown in Fig. 4.

2) The second classifier, **support vector machine** (SVM), will be described and explained in detail later in this section.

It can be easily proven that both k-NN and SVM classifications can be mathematically formulated as optimization problems (Pardalos et al. 2007; Kartelj 2015).

Dimensionality reduction

There are two types of benefits for applying dimensionality reduction by feature selection for the classification process: firstly, by eliminating unnecessary features, it is possible to eliminate dataset noise that degrades the quality of the classification model; secondly, the problem dimension is decreased and the efficiency is increased.

In a biomedical context, dimensionality reduction is very often used: for the analysis of 1H nuclear magnetic resonance spectra from human brain tumor biopsies (Gray et al. 1998), in digital mammography to distinguish benign and malignant microcalcifications (Verma and Zhang 2007), etc. Also, various optimization techniques are used for dimensionality reduction by feature selection: genetic programming (Gray et al. 1998), fractional 0-1 programming (Busygin et al. 2005), neural-genetic algorithms (Verma and Zhang 2007), particle swarm optimization (Inbarania et al. 2014),

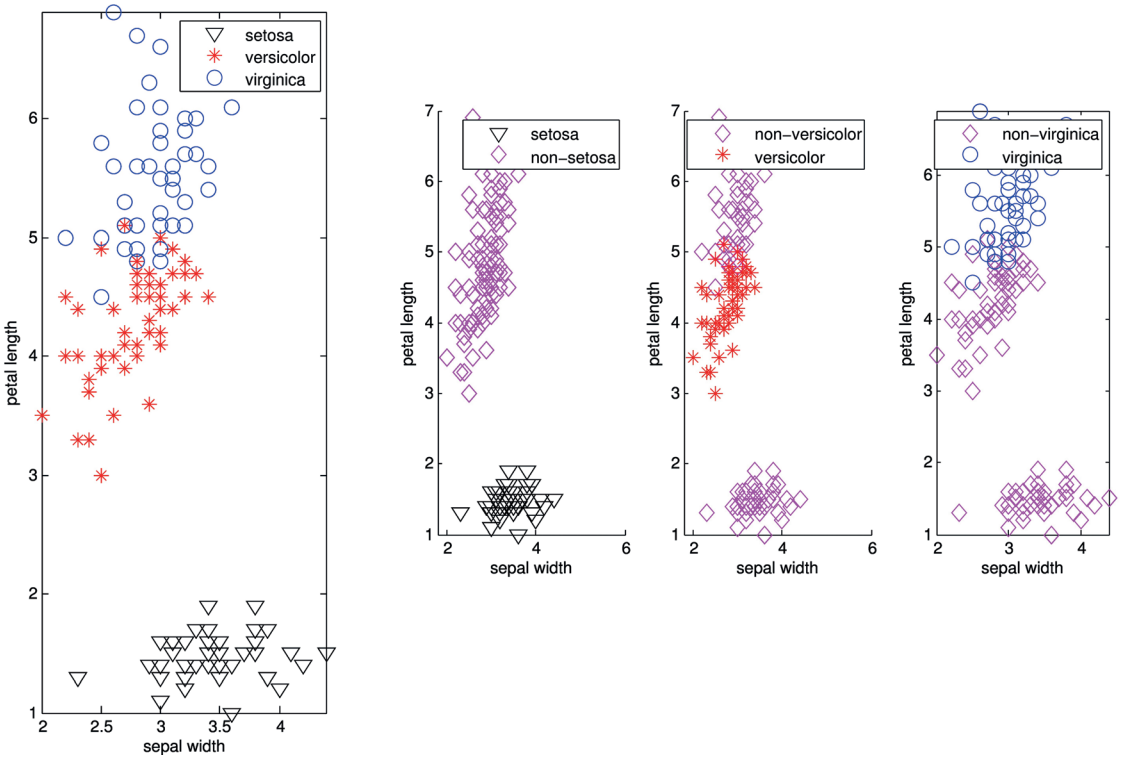

Fig. 3. Multiclass classification.

electromagnetism-like metaheuristics (Kartelj 2015), etc.

The criteria for dimensionality reduction can vary, and in terms of classification problems (which are one focus of this paper) are usually referred to the classification accuracy, model efficiency, level of dimension reduction, or composition of former criteria. Dimensionality reduction algorithms have been utilized as a support for solving various and often real problems. There are three general types of dimensionality reduction algorithms (Kartelj 2015):

1) **wrapper** methods that use a classification algorithm as a black box for the evaluation of feature subsets;

2) **filter** methods that form feature subsets in the preprocessing phase, and do not depend on the employed classification algorithm;

3) **embedded** methods that form a feature subset in the training process and are specific to a given classification algorithm.

This paper considers the first type of dimensionality reduction (the wrapper method), where the 1-nearest-neighbor classifier (1-NN) and support vector machine (SVM) are used as underlying classification mechanisms. In this paper, a study of methods for wrapper dimensionality reduction has been conducted and its characteristics and potentials are discussed.

Support Vector Machine

Support vector machine (SVM) is a supervised machine learning technique used for classification and for the estimation of functional forms in regression problems where it is necessary to predict a continuous variable (Vapnik 1999). SVM uses a training dataset to build a learning function that generalizes well and produces correct predictions when used on unseen data (Kartelj et al. 2013). As with other prediction techniques, it is desirable to check the quality of the learning function on a test set prior to applying it on unseen data.

Here, similarly to k-NN analysis, the binary classification problem is studied. Let the training set, composed of N_{tr} pairs $(\boldsymbol{x}_i, y_i)$, i=1,…, N_{tr} (where $\boldsymbol{x}_i \in R^N$ is an N-dimensional feature vector and $y_i \in \{-1,1\}$ is a corresponding class label) be denoted D_{tr}. SVM employs a training dataset D_{trin} order to find a hyperplane $\boldsymbol{w}\cdot\boldsymbol{x}-b=0$ (w, $x \in R^N$, $b \in R$), which separates feature vectors according to their class labels (Vapnik 1999). Additionally, the separating hyperplane should be maximally distant from the training vectors on both sides (as shown in Fig. 5). In SVM classification, the upper bound for the generalization error is minimized when the distance between vectors and the separating hyper-plane is maximized. The important practical property of the bound is its independency from the dimensionality of the feature space.

SVM is an extremely successful classification method that is extensively applied to problems in biomedicine: in cell death discrimination by Raman spectroscopy (Pyrgiotakis et

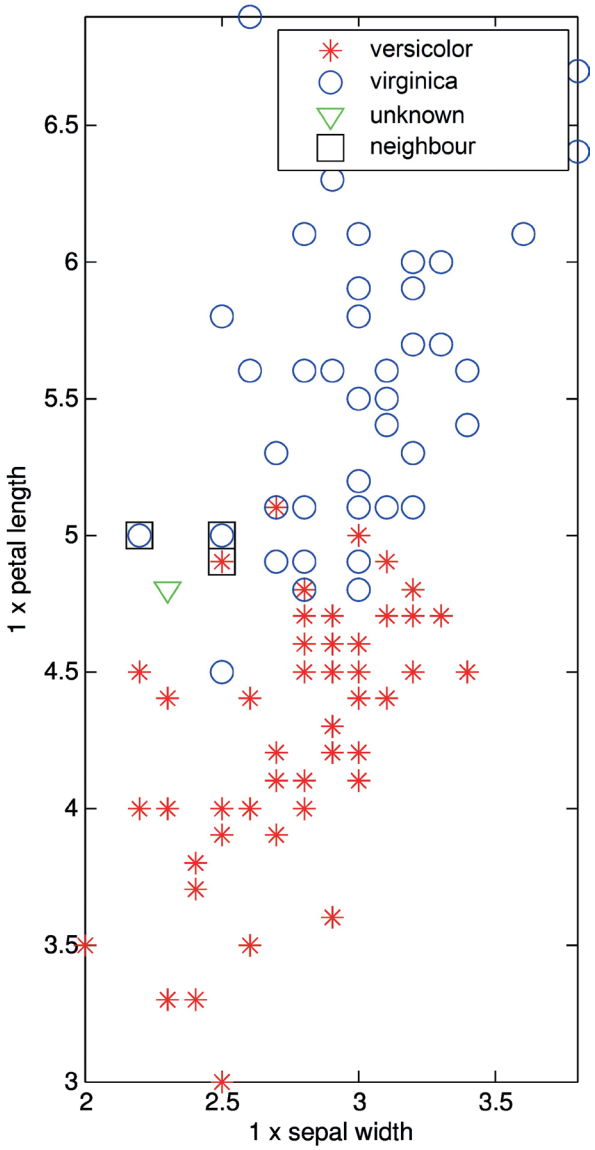

Fig. 4 . kNN classification for k =3.

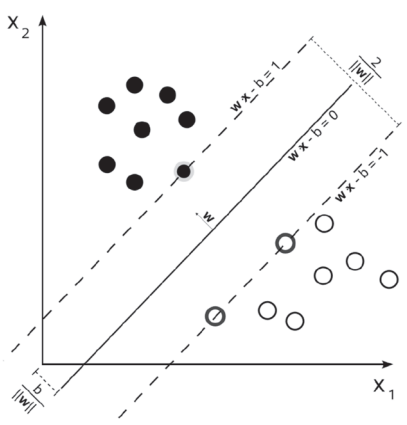

Fig.5. Hyperplane in SVM.

al. 2009), in mortality prediction of septic patients (Vieiraa et al. 2013), to identify and analyze certain post-translational modifications of proteins (Qiua et al. 2017), to classify smoking cessation status (Kartelj 2010), to measure cancer morbidity and mortality data in a cancer registry (Varlamis et al. 2017), etc.

Building a separating hyperplane is not possible when the feature space is not linearly separable, as shown in Fig. 6, which shows one example of a linearly inseparable feature space. In the case described in this figure, the original feature space is mapped to another where linear separation is possible. The space transformation leads to an increase in the dimensionality of a problem, which is shown in Fig. 7, upon applying the following transformation
$F(p, s) = (z_1, z_2, z_3) = (p^2, \sqrt{2}\, ps, s^2)$.

Fortunately, this does not affect the overall performance of SVM, because SVM makes no direct usage of the feature vectors from the mapped space. Instead, SVM employs a similarity function, called a **kernel** function $K : R^N \times R^N \rightarrow R$, which is defined for each pair of feature vectors $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ and represents a similarity (or distance) metric between them. The essential property of the kernel function is that it can be calculated in the original feature space. In this way, increasing the dimensionality of the space is unimportant from the perspective of SVM performance.

It is clear that the quality of the SVM classification depends of the adequate selection of the SVM kernel function. Moreover, it was shown that, beside the type of kernel function, the value of the **regularization parameter** (or penalty parameter) C, which representsthe upper bound for the Lagrange multipliers used in the optimization procedure, is also tightly related to the overall classification error of SVM (Vapnik 1999). The appropriate value of the regularization parameter C is usually determined from alarge positive domain of real numbers.

The most popular models for kernel functions are:

1) The **linear kernel** model: $K(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u} \cdot \boldsymbol{v} = \sum_{i=1}^{n} u_i v_i$ – this is a parameter-free model since it takes only feature vectors and applies the inner product. Thus, the parametrization is concerned only with setting an appropriate value of the SVM regularization parameter C.

2) The **radial basis kernel** model: $K(\boldsymbol{u}, \boldsymbol{v}) = e^{-\sum_{i=1}^{N} \frac{(u_i - v_i)^2}{2\sigma_i^2}}$ – the kernel function itself is parameterized and the parameter set to be tuned is $\{C, \sigma_1, \sigma_2, \ldots, \sigma_N\}$, where σ_i is called a scaling factor that corresponds to a radius of the radial basis function used for the i-th feature.

Sometimes, it is not even enough to use the most appropriate kernel and its underlying parameter structure. This happens in scenarios when training data consists of heterogeneous features, usually grouped in several related clusters of features. Fortunately, a single SVM model can use many kernel functions and hence it is well suited for heterogeneous feature spaces. Each kernel can have its own set of param-

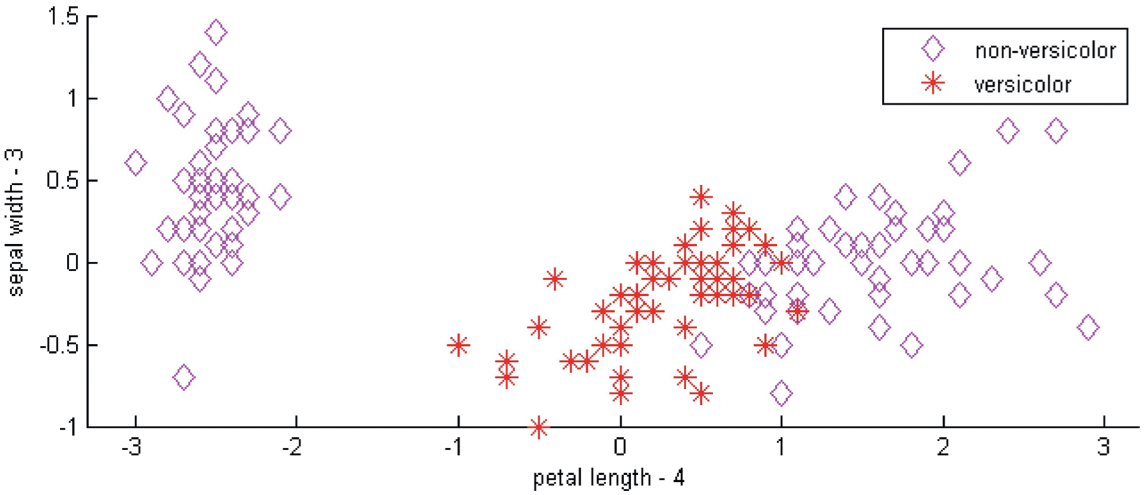

Fig. 6. Linearly inseparable feature space (p, s) for classification.

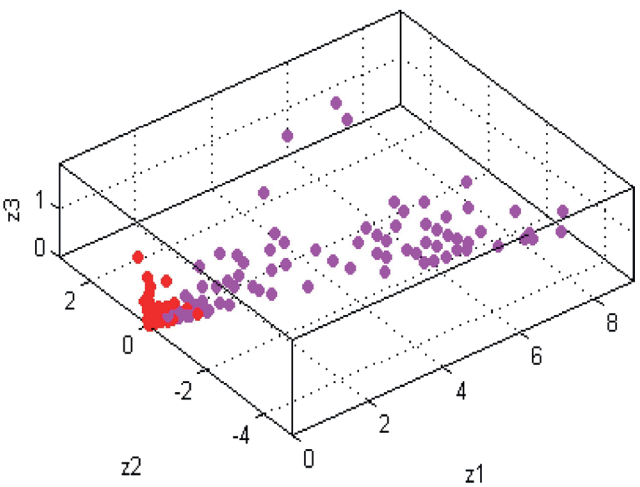

Fig. 7. Same space after transformation **F(p, s)=(z1,z2,z3)=(p^2,$\sqrt{2}ps$,s^2).**

eters that impact the overall prediction quality. In this paper, a method for SVM parameter selection for SVM classifiers with previously defined kernels will be introduced and the obtained results will be presented.

Maximum betweenness problem

The betweenness problem is a well-known optimization problem. For a given finite set S of n objects $S=\{x_1, x_2,\ldots, x_n\}$ and a given set C of triples $(xi, xj, xk) \in S \times S \times S$, the betweenness problem is a problem of determination of the total ordering of the elements from S, such that triples from C satisfy the "betweenness constraint", i.e. the element x_j is between the elements x_i and x_k (Filipović 2011). The problem presented in this paper, called the maximum betweenness problem (MBP), deals with finding the total ordering that maximizes the number of satisfied constraints.

The MBP, as well as other betweenness problems, belongs to a class of discrete optimization problems. Those problems have important applications in various fields, including bioinformatics. For example, the MBP is used for solving some physical mapping problems in molecular biology (Chor and Sudan 1998). During the radiation hybrid experiments, X-rays are used to fragment chromosomes. If the markers on chromosomes are more distant, the probability that the given dose of an X-ray will break a chromosome is greater. In this way, markers are placed on two separate chromosomal fragments.

By estimating the frequency of the breaking points, and thus the distances between markers, it is possible to determine their order within a chromosome in a manner analogous to meiotic mapping. In this context, improvement of the radiation experiment can be achieved by finding the total ordering of the markers that maximizes the number of satisfied constraints. The software package RHMAPPER (Lincoln 1996; Slonim et al. 1997) uses this approach to produce the order of framework markers by employing two greedy algorithms for solving the betweenness problem.

A detailed analysis of the experiments of radiation hybridization is beyond the scope of this paper and can be examined in Goss and Harris (1975) and Cox et al. (1990).

The number of violations, i.e. the number of triples with unsatisfied betweenness constraints can be penalized by imposing weights. The problem that deals with the minimization of the total sum of these weights is called the weighted betweenness problem (WBP). This problem also appears in computational biology (Christof et al. 1998) – more precisely, in the physical mapping with end probes. In this paper, an electromagnetism-like algorithm for solving the MBP is studied and compared to alternatives.

Electromagnetism-like metaheuristic

Problems that have been described in the previous section (dimensionality reduction, SVM parameter selection and maximum betweenness) will be solved by applying the well-known optimization method of electromagnetism-like metaheuristic (EM), as shown in Fig. 8.

Description of the EM

The electromagnetism-like metaheuristic proposed in Birbil and Fang (2003), represents a population-based optimization technique inspired by mechanisms of interaction among electrically charged particles (called EM points). The method employs a proficient search process governed by EM points, where each of them represents a single candidate solution of the underlying problem. EM points that represent better solutions are awarded with higher charge. This is crucial for leading a search process towards promising solution regions, because the EM points with a higher charge attract other points more strongly. The exact attraction-repulsion relationship is given in formula analogues to Coulomb's Law.

Electromagnetism-like algorithms have turned out to be successful in solving many problems with a practical and theoretical background: in Su and Lin (2011) the EM technique has been adopted to solving feature selection problems. The hybrid algorithm based on EM and simulated annealing is proposed in the paper by Tavakkoli-Moghaddam et al. (2009) for a job shop problem. The EM method for an uncapacitated multiple allocation hub location problem (UMAHLP) has been proposed in Filipović (2011).

The overall structure of the EM algorithm is described in the flowchart in Fig. 9. An EM requires only two control parameters: N_{it} is the number of the main loop iterations, and M represents the number of EM points. The points are first assigned with initial solutions, after which the algorithm enters the main loop. The main loop iterates N_{it} times and within iteration every EM point $\boldsymbol{p}_i$, $i = 1,\ldots M$ is subjected to the objective value calculation, i.e. to measuring the quality of the solution represented by that point.

The next step is the calculation of the EM points' charges. As previously mentioned, the charge of a fixed EM point will depend on its solution quality, according to the formula:

$$= e^{-N\frac{obj(\boldsymbol{p_{best}})-obj(\boldsymbol{p_i})}{\sum_{k=1}^{M} obj(\boldsymbol{p_{best}})-obj(\boldsymbol{p_k})}}$$

where N is dimensionality on EM point space, obj is an objective function and $\boldsymbol{p}_{best}$ denotes the EM point with

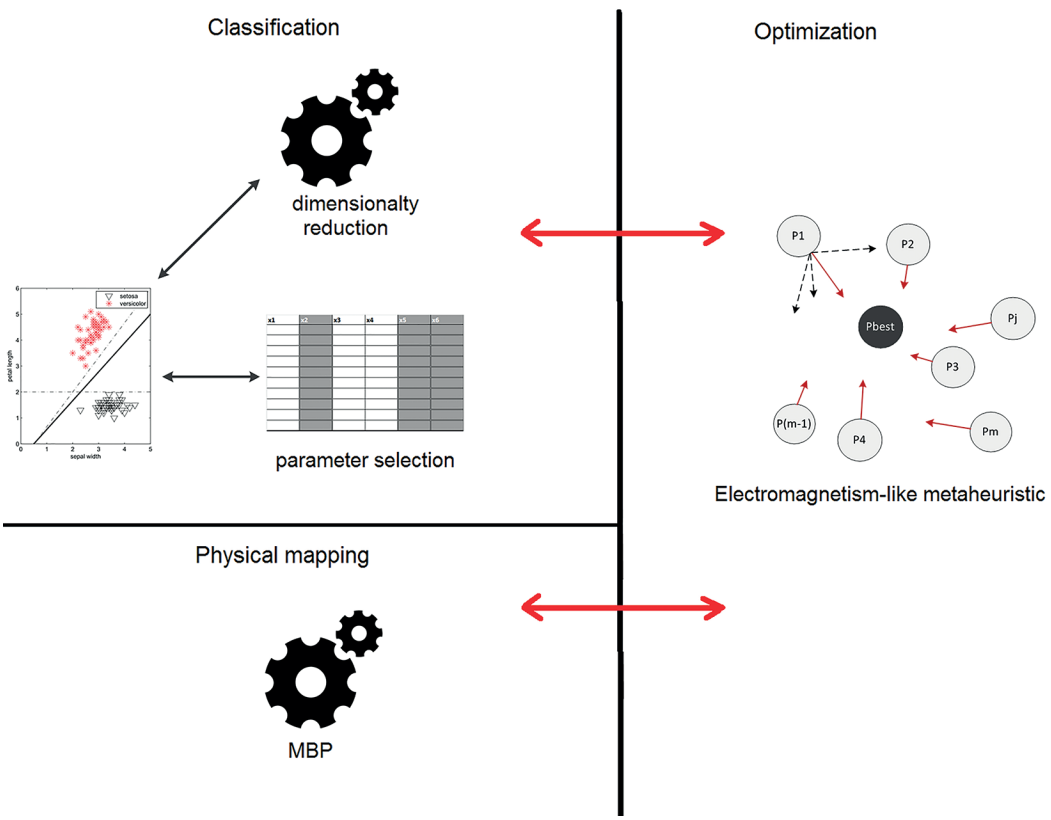

Fig. 8. EM application.

```
input : N_it, M
p = createInitialEMPoints(M);
for iter ← 1 to N_it do
    for i ← 1 to M do
        objFunction(p_i);
    end
    charges(p);
    forces(p);
    relocateEMPoints(p);
end
printSolution();
```

Fig. 9. Outline of the EM method

the highest objective value. An illustrative example with EM points and their calculated charges is shown in Fig. 10 (A).

After all of the charges are calculated, the total impact on each point is calculated by superpositioning particle pairwise interaction forces, which are calculated using the following formula:

$$\boldsymbol{F_i} = \begin{cases} \sum_{j=1,j\neq i}^{M} (\boldsymbol{p_j} - \boldsymbol{p_i}) \dfrac{q_j \times q_i}{\|\boldsymbol{p_j} - \boldsymbol{p_i}\|^2}, & p_j^{obj} < p_i^{obj} \\ \sum_{j=1,j\neq i}^{M} (\boldsymbol{p_i} - \boldsymbol{p_j}) \dfrac{q_j \times q_i}{\|\boldsymbol{p_j} - \boldsymbol{p_i}\|^2}, & p_j^{obj} \geq p_i^{obj} \end{cases}.$$

It should be noted that forces are calculated similar to Coulomb's Law in the sense that the force between each two particles is proportional to the product of their charges and inversely proportional to their distances. Fig. 10 (B) shows the calculated force vectors for every EM point given in the previous example.

After calculating all of the forces $\boldsymbol{F_i} = (F_i^1, F_i^2, \ldots, F_i^N)$ i=1,…M, the movement procedure is applied. The movement of each EM point $\boldsymbol{p_i} = (p_i^1, p_i^2, \ldots, p_i^N)$ is guided by the direction and magnitude of corresponding force vector $\boldsymbol{F_i}$. The following formula determines the movements of EM points:

$$p_i^k = \begin{cases} p_i^k + \lambda \dfrac{F_i^k}{\|\boldsymbol{F_i}\|}(1 - p_i^k), & F_i^k > 0 \\ p_i^k + \lambda \dfrac{F_i^k}{\|\boldsymbol{F_i}\|} p_i^k, & F_i^k \leq 0 \end{cases}$$

An illustrative example that describes movements of EM points whose charges and forces are calculated is shown in Fig. 10 (C).

The remainder of this section elaborates how the previously described EM can be modified for the dimensionality reduction problem, for the SVM parameter selection problem and for the MBP.

EM for dimensionality reduction

An EM metaheuristic for dimensionality reduction is designed as an electromagnetism-like optimization method, with some specific aspects that depend on the dimensionality reduction problem (Kartelj 2015). Four aspects that make a difference between the EM for dimensionality reduction and other EM methods will be explained in detail.

The first distinctive characteristic is the calculation of an **objective function**: all EM points $\boldsymbol{p_i}$, i=1,…M are converted from the real-valued vectors to corresponding binary vectors $\boldsymbol{s_i}$, i=1,…M in the following way: $s_i^k = \begin{cases} 0, & p_i^k < 0.5 \\ 1, & p_i^k \geq 0.5 \end{cases}$, where 1/0 indicates whether the feature is included or not. The objective function is calculated in two ways depending on the type of classifier:

(a) if the 1-NN classifier is used, a 5-fold cross-validation is performed and the objective value is calculated as the balanced classification accuracy;

(b) if the SVM classifier is employed, a 2-fold cross-validation is performed, while the objective value is calculated as classification accuracy, i.e. the average percentage of correctly predicted records.

The second distinctive characteristic is use of the **local**

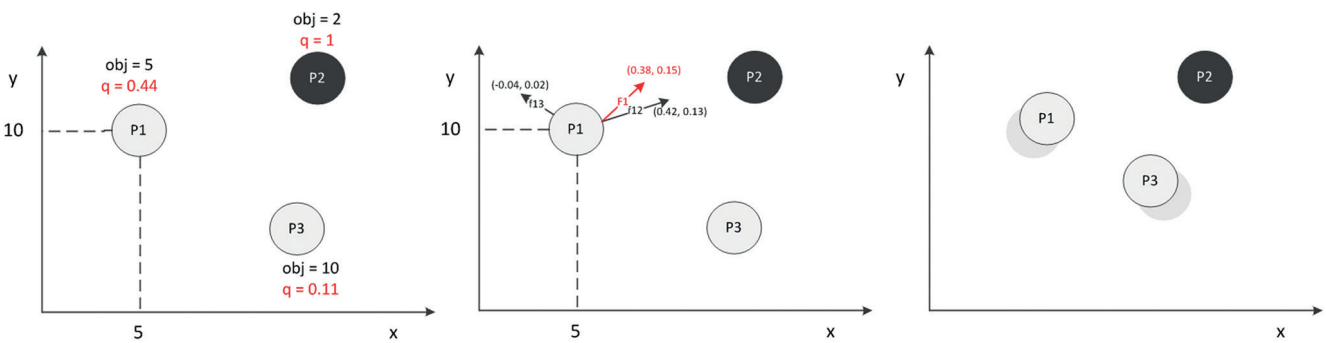

Fig. 10. EM - (A) calculation of the charges, (B) calculation of the forces, (C) movements of the EM points.

search (LS) procedure. After calculating the objective value for all EM points, the local search procedure is applied to at most one of them. At the beginning, the EM for dimensionality reduction examines whether a local search (LS) procedure needs to be invoked. The reasons for this examination are twofold:

(a) LS procedures are usually time consuming, and any reduction in LS calls can (often significantly) decrease the algorithm execution time;

(b) precisely tuned criteria as to whether the LS is or is not called can increase the exploratory properties of the candidate solution space, directing the search process into potentially successful unexplored regions.

Therefore, in this EM method, the LS procedure is called only if the conjunction of the following conditions is satisfied:

(a) The EM point has the best, or the second best objective value;

(b) LS has never been applied to the current EM point, or its objective value has been changed since the last application of LS;

(c) The best objective value has not been changed for at least 10 generations;

After that, if the criteria for invoking the LS on the current point are satisfied, LS is performed. LS consist of two procedures: the first is 1-swap LS with immediate application of improvement, and the second is 2-swap LS with application of the best-found improvement.

In the **1-swap** procedure, a single bit is changed, including/excluding the corresponding feature in/from the solution. If an improvement is detected, it is immediately applied and the LS continues with the new and improved point. Procedure 1-swap stops when no new improvement can be found.

The second **2-swap** procedure consists of the following steps: first, the algorithm excludes from the solution a certain feature; after that, the algorithm searches for the feature outside of the solution, which, when included, produces the improvement. The excluded feature is then compared to every additional feature, not originally included in the solution, and the one providing best improvement is being swapped with the excluded feature. Procedure 2-swap stops when all features belonging to the solution have been examined for exclusion.

The third distinctive characteristic is **scaling**: when the EM point coordinates (related to the features) have values close to 0.5, unnecessary dispersion of the search can appear, because the crossing of the threshold value occurs too frequently. In such cases, the decision as to whether the corresponding features are or are not to be included in the solution changes too often, the convergence of the algorithm is weakened and the overall search process becomes unreliable.

In order to avoid this, the standard EM method is extended by introducing the specific scaling procedure which enables better control of the movements of EM points: $\boldsymbol{p}_i = \alpha\, \boldsymbol{s}_i + (1-\alpha)\, \boldsymbol{p}_i$, where scaling factor α is a number between 0 and 1.

The fourth distinctive characteristic is **caching**: prior to calculation of the objective function, vector is looked-up in the cache collection structure. Only if it is not found is the 1-NN or SVM classifier called to calculate the corresponding objective function. Otherwise, the objective value is taken from the cache collection.

The described method will be compared with various algorithms for dimensionality reduction found in the literature: the EM algorithm proposed in Chao-Ton and Hung-Chun (2011), the genetic algorithm from Yang and Honavar (1998), and with two variants of the particle swarm optimization method proposed in Li-Fei et al. (2012).

EM for SVM parameter selection

The EM for SVM parameter selection is designed as the EM optimization method, with some specific aspects that depend on the specific problem to be solved – SVM parameter selection. This subsection describes elements that make the difference between EM for SVM parameter selection and other EM methods (Kartelj et al. 2013).

Electromagnetism-based metaheuristic for SVM parameter selection is schematically described in Fig. 11. Elements that make this method distinctive are:

1) Calculation of the objective function value, which reflects the quality of the solution represented by the EM point. The natural choice for the objective function is estimation of the generalization error of the SVM classifier. The straightforward estimation of the generalization error is the classification error on the training set. Besides this approach, other more subtle measures have been proposed. For example, leave-one-out (LOO) estimation is performed by removing each training vector from the training set, building a classifier and then testing it on the removed vector. The overall estimation of the classification error is finally calculated as the sum of errors for all removed training vectors. Relaxation of LOO is a k-fold cross-validation-based estimation of the expected generalization error. It is calculated by splitting the training set into k folds and using each fold as a validation set, while the remaining folds are used for the learning phase. After k iterations, when each fold is used once for validation, error estimation is calculated as an average error across k validation sets. The first two steps transform EM point coordinates to SVM parameters and use these parameters to calculate kernel matrix values. After that, the objective function value is calculated with the cross-validation technique. For small problem test instances (datasets) with homogenous (non-clustered) features, 5-fold cross-validation is employed, and for large datasets the classification error estimation is based on the 5×2-fold cross-validation.

2) Application of LS techniques, which is done after

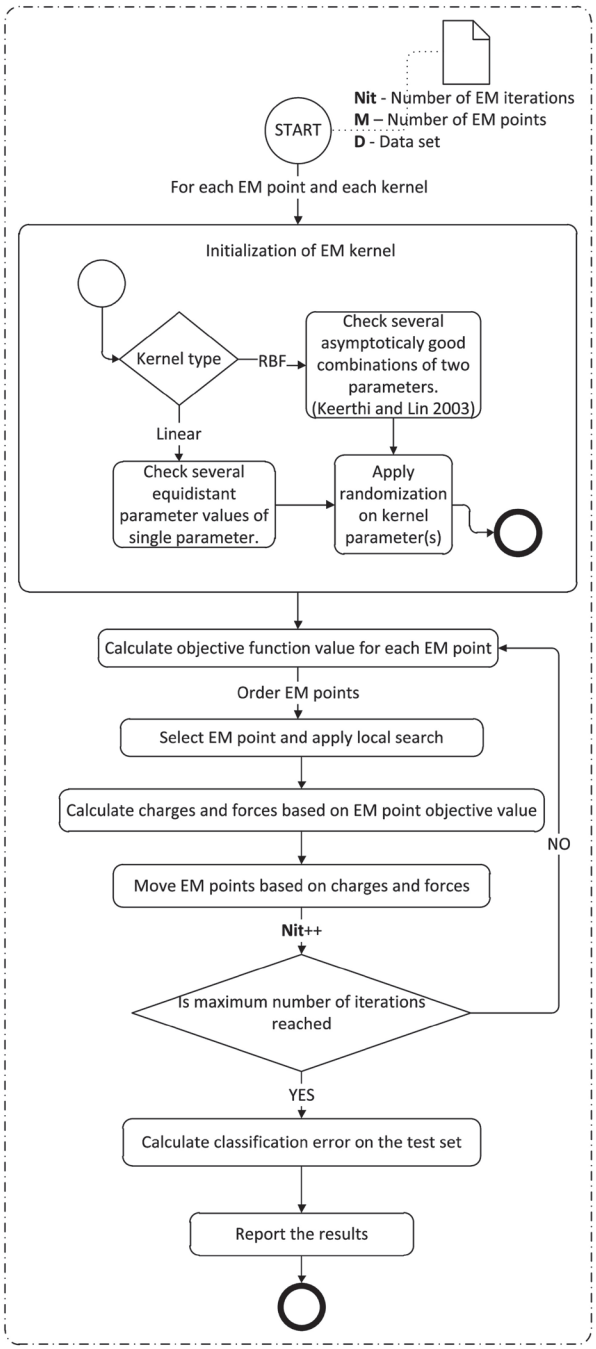

Fig. 11. Flowchart of EM method for SVM parameter selection.

calculation of the objective value for all EM points. As in the previous case, LS separates into two phases – the first is the selection of the point at which the LS will be applied, and the second is the application of LS. Regarding the first phase, candidates for LS application are chosen similarly to the previously described procedure: the EM points are first sorted in ascending order with respect to the classification error estimation, then checked if the EM point fulfills the necessary requirements (whether the LS has never been applied to this point before, or it has been applied, but the value has changed since the last application of LS). The second phase of the procedure is a search for an improvement of the actual parameter setting. The algorithm attempts to find an improvement in both directions for every coordinate of the selected EM point: left (value 0) and right from the boundary value (value 1). This is performed by increasing the value of the coordinate by 1/10 of the remaining interval on the left and right sides of the current coordinate value. When an improvement is found, it is immediately applied and a search for a new improvement is performed in the same direction. If an improvement is not found, the search process continues in the opposite direction once. After all encoded kernel parameters have been checked for improvement, the algorithm ends.

3) Scaling, described in the previous subsection.

4) Caching, described in the previous subsection.

The described method will be compared with other algorithms for SVM parameter selection available in the literature: the method presented in Keerthi and Lin (2003), the ant colony optimization (ACO) method described in Zhang et al. (2010), the evolutional strategy (ES) method of Phienthrakul and Kijsirikul (2010), and the VNS method proposed in Carrizosa et al. (2012).

EM for MBP

Elements of the specific EM for MBP (Filipović et al. 2013) are as follows:

1) Objective function evaluation. In order to maintain search effectiveness of the algorithm, the choice of an appropriate representation of the candidate solution plays a key role. In the case of MBP, each EM point in the solution set is related to one ordering of the set $S = \{1, 2,\ldots, n\}$. This ordering is used to determine the number of satisfied constraints in the objective function. Let the EM point be represented as an n-dimensional vector of real valued coordinates, taking values from [0,1] and denoting that vector as $\boldsymbol{p} = (p_1, p_2,\ldots, p_n)$, where $p_i \in [0,1]$, $i = 1,\ldots, n$. For a given EM point $\boldsymbol{p}$, each element of the set S corresponds to one coordinate of that point and vice versa. The point p determines the corresponding ordering relation: if i and j are two elements from S, then $i \lessdot j \leftrightarrow p_i < p_j$. Now the objective function that calculates the total number of satisfied constraints can be introduced. If all coordinates p_i are different, then the ordering defined by the EM point $\boldsymbol{p}$ induces a 1-1 function f:S→S, which is actually a permutation of the set S. This function f is determined by sorting the set of indices of the point p by the criteria determined by the ordering relation: if i and j are two indices, then $i \lessdot j \leftrightarrow p_i < p_j$.

2) Combination of LS and caching. The algorithm tries to improve each EM point within the iteration. This is accomplished by a special local search procedure that combines

the 1-swap local search approach and caching technique. Both 1-swap and caching are explained in previous subsections.

3) Scaling has already been described earlier in this section.

This EM-like method will be compared with metaheuristics for solving the MBP: genetic algorithms with and without LS (Savić et al. 2011). Moreover, the exact integer programming-based solver CPLEX (IBM Corporation 2016), which is capable of dealing only with smaller problems, will be used for verification and comparison.

Application of the EM

Each of the following subsections contains results for one of the considered problems: dimensionality reduction, SVM parameter selection and maximum betweenness. The results were obtained by executing experiments on test instances with a biomedical/biological origin (various classification datasets from the UCI repository and data extracted from the RHMAPPER software package).

EM applied to dimensionality reduction problem

The method is implemented in the C programming language and compiled with Visual Studio 2010 compiler. All the tests are executed on a PC with 2.4GHz Intel processor and 4GB RAM under the Windows 7 operating system.

Two experiments were conducted on two separate collections of classification datasets with biological/biomedical origin taken from the UCI machine learning repository (Lichman 2013).

The first collection, consisting of 6 datasets (described in Table 1), is used for comparison. The structure of the table

Table 1. DR – Classification datasets used in first xperiment.

dataset	*N*	N_c	N_r
abalone	8	11	3842
iris	4	3	150
water	38	4	513
wine	13	3	178
wisconsin	9	2	683
yeast	8	9	1479

is as follows: dimensionality (e.g. number of features, denoted as N), number of classes (N_c) and dataset size (N_r). The previously described EM for dimensionality reduction is compared with the EM algorithm (Chao-Ton and Hung-Chun 2011) denoted as EMstc, and with the genetic algorithm (Yang and Honavar 1998), denoted as GA.

Experiment 1: In order to compare method EM with EM^{stc} and GA, the accuracy and feature reduction were calculated on the 1-NN classifier. For each dataset, the EM algorithm was executed 10 times. Each execution used a different random seed that consequently produced different fold partitioning. For each dataset, the average values of RF are recorded. The number of iterations N_{it} and the number of EM points M were kept uniform across all datasets: N_{it} = 600, M = 150. The parameter α which controls the scaling intensity was set to 0.1.

Table 2 shows comparison results. The first two columns of the table are the dataset name and average values obtained by EM^{stc} and *GA* (measured in percentages), respectively. The next column *FS* shows the average optimal solution obtained by the full search algorithm after 10 executions. The column *EM* takes value *opt* if the obtained average solution is equal to the average optimal solution of FS, which means that in such cases EM obtains optimal solutions in every of 10 executions. The last three columns present the reduction rate (in percentages) for the three algorithms that are compared.

The results in Table 2 suggest that EM outperformed the other methods in 4 out of 6 cases in terms of feature reduction rate. It can also be seen that the EM obtained an optimal average solution in 5 out of 6 cases, meaning that the success rate was 83.3% for these datasets.

Table 3 shows the execution times (in seconds) of the compared methods. Five datasets were solved easily with FS, all in less than 10 seconds. The studied EM proved to be similarly fast on these small datasets, which is the consequence of caching.

Experiment 2: In the second experiment, the EM method was compared to the PSO^1 and PSO^2 methods. The comparison was based on three datasets from the UCI repository. LIBSVM implementation of SVM (Chih-Chung and Lin 2011) was used as an underlying classification algorithm. The error cost parameter of SVM was set toand the radial basis function with was used as a kernel. For multiclass datasets, a one-against-rest strategy was used. The maximal number of EM iterations was set at. As a resampling technique, 2-fold cross-validation was used. The reported values of classification accuracy and percentages of retained features are all reported as averages after running the EM algorithm times for each dataset.

The comparison is based on the average classification accuracy and the average percentage of retained features.

Table 4 shows the following information: dataset name, number of features (N), number of classes (N_c), classification accuracy of the first and second variant of the PSO algorithm (PSO^1 and PSO^2), optimal classification accuracy obtained by the full search algorithm (if it finishes execution - *FS*), the EM classification accuracy (*EM*), and finally the average numbers of retained features ($PSO^1{}_d$, $PSO^2{}_d$ and EM_d).

The EM reached all average optimal solutions on smaller datasets where the FS algorithm finished its execution. In the remaining large dataset, FS algorithm execution lasted more than 3 days, and it was terminated before completion.

Table 2. DR -Accuracy and feature reduction comparison for 1-NN classifier.

dataset	EM^{stc}	GA	FS	EM	EM_{RF}^{stc}	GA_{RF}	EM_{RF}
abalone	24.35	24.37	23.99	***opt***	52.50	50.00	**57.50**
iris	98.00	98.00	99.39	***opt***	55.00	**60.00**	50.00
water	73.34	66.28	-	**80.03**	54.21	47.89	**63.16**
wine	98.57	98.57	99.80	***opt***	58.46	61.54	**72.31**
Wisconsin	98.25	98.04	98.62	***opt***	**53.33**	40.00	48.89
yeast	47.07	47.03	51.15	***opt***	17.50	12.50	**22.50**

Table 3. DR - Computational times for 1-NN classifier.

dataset	***FS_t (s)***	***EM^{stc}_t (s)***	***GA_t (s)***	***EM_t (s)***
abalone	8.4	1376.1	7097.2	**7.2**
iris	**0.0**	7.5	288.6	0.9
water	>3 days	262.8	1574.5	**55.7**
wine	**1.9**	153.0	269.9	2.5
Wisconsin	**0.6**	70.6	2096.8	2.0
yeast	**1.2**	234.7	2252.3	2.4

For comparison it is better if the number of retained features is smaller. It can be noted that the EM reached the smallest average number of features in all datasets.

EM applied to SVM parameter selection

In this section, the performances of the studied EM approach were evaluated in three experiments that were made using three collections of small and medium-sized datasets with up to 60 features. Table 5 contains information about the 6 datasets that were used in the third and the fourth experiments. It consists of the following columns: dataset name (*dataset*), number of features (N), number of training samples (N_{tr}) and number of testing samples (N_{ts}). The datasets are available on the Machine Learning UCI repository (Lichman 2013). Table 6, with the same structure as the previous one, contains information about the 5 datasets that were used in the fifth experiment.

Due to homogeneity, i.e. the absence of clusters of features in the first experimental collection, the single-kernel RBF model for SVM was adopted. The search process of the EM algorithm was guided by the classification error estimate, calculated as a 5-fold cross-validation classification error. At the end of the search process, when the termination criterion was met, the SVM parameters encoded by the best EM point were used to train the prediction model. The model was then applied to the testing set and the obtained test error was subsequently used for comparison with the other methods.

The EM algorithm was written in the C programming language and Visual Studio 2010 compiler was used for compilation. All tests were carried out on the Intel Xeon E5410 @ 2.34GHz under Windows 7 operating system.

Experiment 3: Concerning EM parameters, in this experiment the setting used is $M = 5$ and $N_{it} = 5$.

Table 7 contains the results from the third experiment, i.e. a comparison of the EM method and two other methods described in the literature. Columns denoted by *KL* and *VNS* refer to the results reported in Keerthi and Lin (2003) and Carrizosa et al. (2012) after a single execution. The number of objective value evaluations, running time (in seconds), and the iteration in which the solution was found are also presented in the last three columns of the table, denoted by $Eval_{EM}$, t_{EM} and $Iter_{found}$ respectively.

The results indicate that the studied EM method outperformed the other two approaches in 4 out of 6 testing benchmarks, it shared the same (best) result on one data set and produced the second-best solution on the remaining one.

Experiment 4: For this experiment, the parameters for the EM were $M = 8$ and $N_{it} = 1000$. The results from the fourth experiment are shown in Table 8. The first three columns show the classification errors of SVM tuned by grid search (*GS*), ant colony optimization (*ACO*) and the studied electromagnetism-like algorithm (*EM*). Both the grid search and ant colony optimization results were taken from Zhang et al. (2010). Finally, the corresponding computational times (in seconds) are shown in the last three columns (t_{GS}, t_{ACO} and t_{EM}).

It is evident that the EM outperformed both comparison algorithms on all tested datasets. It can also be seen that the computational times differed significantly, i.e. the EM usually spends less time. This is due to the fact that the EM and ACO have different finishing criteria.

Experiment 5: In this instance, in order to make a fair

Table 4. DR - Accuracy and number of features comparison table for SVM classifier.

dataset	***N***	***N_c***	***PSO^1***	***PSO^2***	***FS***	***EM***	***PSO^1_d***	***PSO^2_d***	***EM_d***
breast-cancer	30	2	96.83	**97.66**	-	95.92	11.1	12.2	**6.4**
heart	13	2	84.30	86.01	83.74	**opt**	8.6	7.5	**3.5**
wine	13	3	99.19	99.72	97.30	**opt**	8.3	8.6	**6.7**

Table 5. PS –Datasets used in the third and partially in the fourth experiment.

dataset	N	N_{tr}	N_{ts}
banana	2	400	4,900
diabetes	8	468	300
heart	13	170	100
splice	60	1,000	2,175
thyroid	5	140	75
breast cancer	9	200	77

Table 6. PS –Datasets used in the fifth experiment.

dataset	N	N_{tr}	N_{ts}
breast-cancer	10	559	140
Cleveland-heart	13	216	54
Indians-diabetes	8	614	154
liver-disorders	6	276	69
spiral	2	465	117

Table 7. PS - Single RBF kernel on datasets from the third experiment.

dataset	KL	VNS	EM	$Eval_{EM}$	t_{EM} (s)	$Iter_{found}$
banana	11.59	11.61	**11.57**	57	7.35	1/5
breast cancer	29.87	**28.57**	**28.57**	41	4.88	2/5
diabetes	24	24.67	**23.33**	69	9.65	1/5
heart	21	20	**19**	35	0.61	1/5
splice	10.53	**9.93**	10.16	49	39.34	3/5
thyroid	5.33	5.33	**4**	47	0.52	2/5

comparison, the parameters for the EM were set at $M = 100$ and $N_{it} = 100$. The results from this experiment are presented in Table 9. The first column is the dataset name, the second and third columns refer to the average classification error of the grid search (*GS*) and (*ES*), as described in Phienthrakul and Kijsirikul (2010). The average results of the previously described EM method are presented in the fourth column. The last three columns show the number of the objective function calculations ($Eval_{EM}$), the running time (t_{EM} - in seconds) and the average iteration number when the solution was found ($Iter_{found}$).

The results show that the EM algorithm consistently produced the best solutions in all cases.

EM applied to MBP

In this subsection, the computational results of the EM method are presented and discussed. The EM implementation was implemented in the C programming language using Visual Studio 2010 programming environment. All tests were carried out on Intel Xeon E5410, @2.34 GHz.

Experiment 6: The instance group named SAV contains instances that are described in Savić et al. (2011). The set of SAV instances contains a total of 22 problems. The instances have various numbers of elements in set *S* (*N* = 10, 11, 12, 15, 20, 30, 50), and various numbers of triples in *C* (ranging from 20 to 1000). The name of each instance reflects the problem dimension: for example, the instance "11-100" indicates that set *S* has 11 elements and that collection *C* has 100 triples from set *S*. The results obtained by this EM were compared to the results obtained by the GAs (with and without local search), as proposed in Savić et al. (2011).

Because experiments reported in Savić et al. (2011) repeated the executions 20 times, the same scheme was used here: for each instance, the algorithm was run 20 times with different random seeds. For this set of instances, the stopping criteria was set on a maximum of 100 iterations reached, or 20 iterations without changing the best solution. Concerning the value that represents the number of EM points *M*, for all instances (except the largest one) 20 EM points were used, and for the largest one 50 EM points were used.

Table 10 provides the results of this experiment, obtained by the previously described EM on SAV instances. The columns in the table are as follows: the first three contain the instance name, the optimal solution (if it is known) and the best-known solution from the literature (in cases when the optimal solution is not known); the best solution obtained by the EM in 20 runs is given in the fourth column (named EM_{best}); the average running time used to reach the final EM solution for the first time (denoted as *t*) is given in the fifth column; the sixth and seventh columns (marked as t_{tot} and $iter_{LS}$) contain the average total running time and the average number of LS steps for finishing the EM, respectively.

The results shown in Table 10 clearly indicate that the EM reached all known optimal solutions, except one. For all other instances, the EM algorithm achieved the same or better results than the current best-known, except for two instances.

The computational time for smaller instances (up to 20

Table 8. PS - Single RBF kernel on datasets from the fourth experiment.

dataset	GS	ACO	EM	t_{GS} (s)	t_{ACO} (s)	t_{EM} (s)
breast cancer	25.97	25.97	**23.38**	2,547.3	1,437.8	**270.44**
diabetes	23.33	23	**22.67**	29,078	19,298	**1,837.46**
heart	19	16	**15**	1,446.4	519.58	270.71
thyroid	4	2.67	**1.33**	702.89	666.2	**163.43**

Table 9. PS - Single RBF kernel on datasets from the fifth experiment.

dataset	GS	ES	EM	$Eval_{EM}$	t_{EM} (s)	$Iter_{found}$
breast-cancer	13.59	5.44	**3.86**	11,526	663.49	25.2/100
Cleveland-heart	44.44	21.85	**14.81**	10,622.8	132.02	9.4/100
Indians-diabetes	35.03	26.7	**22.4**	11,413.8	1,749.21	52.4/100
liver-disorders	38.26	33.33	**26.96**	11,510.8	383.47	32/100
spiral	**0**	**0**	**0**	11,543.4	603.2	44.4/100

Table 10. MBP – Results of EM in the sixth experiment.

SAV instance	Opt	Best	EM_{best}	t (s)	t_{tot} (s)	$iter_{LS}$
10–20	16	16	16	0.0017	0.03675	2335.4
10–50	29	29	29	0.00505	0.06695	2511.3
10–100	50	50	50	0.01515	0.13505	2901.3
11–20	14	14	14	0.0014	0.0325	2138.4
11–50	33	33	33	0.02135	0.0968	3765.4
11–100	55	55	55	0.0137	0.16275	3505.4
12–20	17	17	17	0.0056	0.0415	2843.6
12–50	34	34	34	0.02605	0.106	4111.5
12–100	56	56	56	0.0366	0.20255	4153.8
15–30	26	26	26	0.01965	0.0805	4272
15–70	–	46	46	0.04615	0.1978	5493.4
15–200	–	106	106	0.21425	0.71025	7139.9
20–40	37	37	37	0.0478	0.16255	6424.5
20–100	–	67	67	0.22035	0.5486	9759.6
20–200	–	116	116	0.38635	1.2053	10
30–60	55	55	54	0.14755	0.4432	11
30–150	–	111	111	0.6347	1.58475	16
30–300	–	185	185	1.3495	3.82755	19
50–100	–	87	87	0.65595	1.7709	20
50–200	–	153	153	2.0437	4.9498	29
50–400	–	265	259	4.32465	12.205	34
50–1000	–	536	536	67.0349	133.796	141

elements and up to 200 constraints) was less than 1 second for all instances. For medium and large-scale instances, the computational time was less than 12 seconds, with the exception of the largest instance for which the computational time was about 130 seconds. The longer execution time for the largest instances was expected because a larger number of EM points were created and maintained (50 instead of 20).

Table 11 contains the results of the comparison among optimization methods: CPLEX, GA, GA+LS and EM. For each of the methods, the solution (denoted as *sol*), or the best and average solutions (denoted as *best* and *avg*) and the execution time (*t*) are displayed.

Table 11 shows that the EM outperformed all other approaches on medium and large instances, comparing the best solutions. Comparing the obtained average best solutions, it can be seen that EM outperformed the other two methods in all instances.

The computational times of the EM approach are also comparable to other methods, especially with GA with LS. For large instances, the GA approach without LS was faster than the EM, but the EM provided significantly better results.

Experiment 7: Experiments were extended by using real problem instances, named REAL, as in Slonim et al. (1997). To gather these instances, the RHMAPPER software package (a tool for creating genome maps developed at the Whitehead Institute/MIT Center for Genome Research) was used. Inside the software distribution package, there is a set of markers from chromosome 18, as well as a complete set of mapped markers from the Whitehead's May 1996 release. This set of markers and the RHMAPPER command were used to find triples to generate triples of markers. A total of 9 problem instances were collected to solve using the previously described EM algorithm.

In Slonim et al. (1997), the focus was to determine the total ordering relation between markers in order to find the path of markers of maximal length. To address the problem, the authors developed an algorithm for solving the variant of the MBP. After the algorithm was executed, the markers that did not conform to the total ordering relation were removed and the path of maximal length based on the satisfied betweenness constraints was established.

This experiment used the same settings for the EM as the previous one: a maximum of 100 iterations reached or 20 iterations without changing the best solution, 20 EM points were used for all instances except the largest one, and 50 EM points for the largest one.

The obtained results are shown in Table 12,which has the same structure as Table 10: the first three columns contain the instance name, also indicating the problem's dimension, optimal solution, if it is known, and the best-known solution from the literature in cases where the optimal solution is not known; the best solution obtained by EM in 20 runs is given in the fourth column; the average running time used to reach the final EM solution for the first time is given in the fifth column; the sixth and seventh columns contain the average total running time and the average number of local

Table 11. MBP - Comparative results and running times in the sixth experiment.

SAV instance	**CPLEX**		GA			GA+LS			EM		
	sol	t(s)	best	avg	t(s)	best	avg	t(s)	best	avg	t(s)
10–20	16	0.437	16	15.75	0.194	16	15.8	0.088	16	16	0.037
10–50	29	7203.8	29	28.95	0.195	29	29	0.09	29	29	0.067
10–100	42	7201.6	50	48.79	0.652	50	48.75	0.116	50		0.135
11–20	14	2.125	14	13.65	0.2	14	13.65	0.089	14	14	0.032
11–50	33	7203.6	33	32.25	0.214	33	32.25	0.095	33	33	0.097
11–100	55	7201.7	55	53.55	0.243	55	53.55	0.115	55	55	0.163
12–20	17	1.156	17	16.6	0.197	17	16.7	0.09	17	17	0.042
12–50	32	7203.8	33	32	0.228	33	32	0.104	34	33.95	0.106
12–100	54	7202.2	56	54.25	0.246	56	54.35	0.119	56	56	0.203
15–30	26	3.172	25	22.75	0.217	25	22.9	0.101	26	24.75	0.08
15–70	45	7202.8	46	43.95	0.231	46	44.15	0.11	46	46	0.198
15–200	98	7201.4	105	102.85	0.289	105	102.85	0.149	106	105.6	0.71
20–40	37	1.625	36	32.3	0.34	37	32.8	0.171	37	35.45	0.163
20–100	63	7201.8	65	62.1	0.398	66	62.9	0.268	67	66.3	0.549
20–200	111	7201.1	113	111.6	0.4	114	111.9	0.225	116	115.05	1.205
30–60	55	7201.8	51	47.75	0.538	53	48.7	0.341	54	52.01	0.443
30–150	105	7200.8	102	95.65	0.627	111	98.5	0.598	111	104.6	1.585
30–300	165	7200.6	173	164.7	0.749	179	167.7	1.002	185	178.1	3.828
50–100	84	7200.9	84	78	1.147	86	81.25	1.163	87	85.45	1.771
50–200	154	7200.4	140	132.1	1.385	151	143.75	3.837	153	147.2	4.95
50–400	225	7200.3	240	230.15	1.535	265	248	7.8	259	252.25	12.2
50–1000	420	7200.2	504	482.9	2.169	532	514.15	19.86	536	524	133.8

search steps for finishing the EM, respectively.

From Table 12 it is evident that the EM easily found the optimal solutions (which were verified by CPLEX) for all of the seven middle-scale instances. The algorithm obtained an optimal solution in all 20 runs. For the two largest real instances, CPLEX could not find an optimal solution in less than 7200 seconds, so that the optimality of the solutions obtained by the EM could not be verified. It is evident that for these two instances, the EM achieved high-quality solutions in a short time, which was less than 55 seconds for the largest instance.

Table 12. MBP - Results of the EM method in the seventh experiment.

REAL instance	Opt	Best	EM (best)	t(s)	t_{tot} (s)	$iter_{LS}$
15–120	118	118	118	0.0042	0.11485	7220.3
16–142	142	142	142	0.00585	0.1682	8537.3
19–187	176	176	176	0.00985	0.2644	9375.1
20–259	257	257	257	0.01765	0.4302	13690.8
24–436	427	427	427	0.0492	1.2161	18088.9
25–305	305	305	305	0.04765	0.85925	17904.3
25–478	477	477	477	0.09525	1.40615	20204.5
33–1310	–	1285	1285	0.6533	8.48835	40408.1
47–2888	–	2785	2785	7.06745	54.7091	77093.6

Conclusions

This paper presents the electromagnetism-like approach for solving various optimization problems. These problems (dimensionality reduction, SVM parameter selection and maximum betweenness) have a great importance in biomedicine and bioinformatics. The described EM metaheuristic uses an adjustable scaling procedure that provides a well-suited control of the optimization process.

Concerning dimensionality reduction, the studied EM method is capable of dealing with the problem of inherent execution time inadequacy of wrapper-based feature selection methods by employing three improvements. The first is an efficiently implemented local search that combines first improvement 1-swap and best improvement 2-swap procedures. The second improvement, which greatly reduces execution time, is the careful application of local search, and the third is using a caching technique. The experimental results obtained on UCI datasets reveal the superiority of the previously described approach as compared to other wrapper approaches with respect to both feature reduction rate and running time.

Concerning SVM parameter selection, the prediction accuracy of the SVM is highly dependent on the values of internal SVM parameters. The traditional approach for solving the problem of parameter setting, grid search, behaves well for the parameter sets of low cardinality. Due to the real-valued nature of parameter domains, the efficiency of the grid search rapidly decreases with the introduction of new parameters. Here, an efficient SVM parameter-tuning algorithm based on the EM is described and compared to its alternatives.

Concerning the MBP problem, the major improvement is an encoding scheme that is used which gives a suitable representation of an individual EM point. This specific encoding scheme enables fast and efficient transformation from the continuous space of EM points to the discrete space of permutations and vice versa, following the idea that minor movements of EM points should not change the objective value. In order to examine the quality of this method, computational experiments are performed on real and artificial datasets and test instances from the literature.

It can be concluded that the EM optimization method can be successfully applied to various subdomains in biomedicine and bioinformatics – from classification to mathematical modelling. An EM should be carefully designed and implemented, primarily taking into account EM point representation, objective function calculation, local search and caching.

Acknowledgments

This research was partially supported by the Ministry of Education, Science and Technological Development, Republic of Serbia, Project No. 174010: Mathematical Models and Optimization Methods for Large-Scale Systems.

References

Allwein E, Schapire R, Singer Y. 2001. Reducing multiclass to binary: a unifying approach for margin classifiers. The Journal of Machine Learning Research.1: 113-141.

Birbil I, Fang, SC. 2003. An Electromagnetism-like Mechanism for Global Optimization. Journal of Global Optimization.25: 263-282.

Busygin S, Oleg P,Pardalos P. 2005. Feature Selection for Consistent Biclustering via Fractional 0-1 Programming. Journal of Combinatorial Optimization.10: 7-21.

Carrizosa E, Martín-Barragán B, Romero Morales D. 2012. Variable neighborhood search for parameter tuning in support vector machines. Technical report. [accessed 28 Apr 2017] http://www.sbs.ox.ac.uk/sites/default/files/SBS_working_papers/Neighborhood_final.pdf.

Chao-Ton S, Hung-Chun L. 2011. Applying electromagnetism-like mechanism. Information Sciences, 181(5): 972-986.

Chih-Chung C, Lin CJ. 2011. LIBSVM: A library for support vector machine. ACM Transactions on Intelligent Systems and Technology.2(27): 1-27.

Choi JW, Lee H, Lee JC, Lee S, Kim YS, Yoon HJ, Kim HC. 2017. Application of genetic algorithm for hemodialysis schedule optimization. Computer Methods and Programs in Biomedicine. 145:35-43.

Chor B, Sudan M. 1998. A geometric approach to betweenness. SIAM Journal on Discrete Mathematics, 11(4): 511-523.

Christof T, Oswald M, Reinelt G. 1998. Consecutive Ones and a Betweenness Problem in Computational Biology. In: Bixby R, Boyd A, Ríos-Mercado R, editors. Integer Programming and Combinatorial Optimization IPCO 98. Berln, Heidelberg: Springer. p. 213-223.

Cox DR, Burmeister M, Price ER, Kim S, Myers RM. 1990. Radiation hybrid mapping: a somatic cell genetic method for constructing high-resolution maps of mammalian chromosomes. Science, 250(4978):245-250.

Filipović V. 2011. An Electromagnetism Metaheuristic for the Uncapacitated Multiple Allocation Hub Location Problem. Serdica Journal of Computing, 5(3):261-272.

Filipović V, Kartelj A, Matić D. 2013. An electromagnetism metaheuristic for solving the Maximum Betweenness Problem. Applied Soft Computing, 13(2): 1303-1313.

Goss S, Harris H. 1975. New method for mapping genes in human chromosomes. Nature, 255(5511): 680-684.

Gray H, Maxwell R, Martínez-Pérez I, Arús C, Cerdán S. 1998. Genetic programming for classification and feature selection: analysis of 1H nuclear magnetic resonance spectra from human brain tumour biopsies. NMR in Biomedicine. 11:217-224.

Grbić M, Kartelj A, Matić D, Filipović V. 2016. Improving 1NN strategy for classification of some prokaryotic organisms. Book of abstracts, Belgrade Bioinformatic Conference (BelBI).

Hammer P, Bonates T. 2006. Logical analysis of data–An overview: From combinatorial optimization to medical applications. Annals of Operations Research. 148:203-225.

Hüffner F, Komusiewicz C, Liebtrau A, Niedermeier R. 2014. Partitioning Biological Networks into Highly Connected Clusters with Maximum Edge Coverage. IEEE/ACM Transactions on Computational Biology and Bioinformatics , 11(3):455-467.

IBM Corporation. 2016. IBM ILOG CPLEX Optimization Studio - CPLEX User's Manual. [accessed 15 Apr 2017]https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.0/ilog.odms.studio.help/pdf/usrcplex.pdf.

Inbarania H, Azarb AT, Jothi G. 2014. Supervised hybrid feature selection based on PSO and rough sets for medical diagnosis. Computer Methods and Programs in Biomedicine. 113:175-185.

Kartelj A. 2010. Classification of Smoking Cessation Status Using Various Data Mining Methods. Mathematica Balkanica. 24:199-205.

Kartelj A, Mitić N, Filipović V, Tošić D. 2013. Electromagnetism-like Algorithm for Support Vector Machine Parameter Tuning. Soft Computing, 18(10):1985-1998.

Kartelj A. 2015. An Improved Electromagnetism-like Method for Feature Selection. Journal of Multiple-Valued Logic and Soft Computing.25(2):169-187.

Keerthi S, Lin, C. 2003. Asymptotic behaviors of support vector machines with gaussian kernel. Neural Computing, 15(7): 1667-1689.

Kononenko I. 2001. Machine learning for Medical Diagnosis: History, State of the Art and Perspective. Artificial Intelligence in Medicine. 23:89-109.

Lavrač N, 1999. Selected techniques for data mining in medicine. Artificial Intelligence in Medicine. 16:3-23.

Lichman M. 2013. UCI Machine Learning Repository. [accessed 11 Apr 2017] http://archive.ics.uci.edu/ml.

Li-Fei C, Chao-Ton S, Kun-Huang C. 2012. An improved particle swarm optimization for feature selection. Intelligent Data Analysis. 16: 167-182.

Lincoln S. 1996. RHMAPPER: An Interactive Program for Radiation Hybrid Mapping. [accessed 29 Apr 2017] ftp://ftp.broad.mit.edu/pub/software/rhmapper/HEADER.html~.

Pardalos P, Boginski V, Prokopyev O, Suharitdamrong W, Carney P, Chaovalitwongse W, Vazacopoulos A. 2005. Optimization Techniques in Medicine. In: Audet C, Hansen P, Savard G, editors. Essays and Surveys in Global Optimization. New York: Springer.p.211-232.

Pardalos P, Boginski V, Vazacopoulos A, editors. 2007. Data Mining in Biomedicine. New York: Springer.

Pardalos P, Romeijn E. 2009. Handbook of Optimization in Medicine. New York: Springer.

Patel V, Shortliffe EH, Stefanelli M, Szolovits P, Berthold MR, Bellazzi R, Abu-Hanna A. 2009. The coming of age of artificial intelligence in medicine. Artificial Intelligence in Medicine. 46:5-17.

Phienthrakul T, Kijsirikul B. 2010. Evolutionary strategies for hyper-parameters of support vector machines based on multi-scale radial basis function kernels. Soft Computing, 14(7): 681-699.

Pyrgiotakis G,Kundakcioglu O, Finton K, Pardalos P, Powers K, Moudgil B. 2009. Cell Death Discrimination with Raman Spectroscopy and Support Vector Machines. Annals of Biomedical Engineering, 37(7): 1464-1473.

Qiua WR, Sun BQ, Tang H, Huang J, Lin H. 2017. Identify and analysis crotonylation sites in histone by using support vector machines. Artificial Intelligence in Medicine [accessed 25 Apr 2017]; http://www.aiimjournal.com/article/S0933-3657(16)30582-6/fulltext . doi: http://dx.doi.org/10.1016/j.artmed.2017.02.007.

Savić A, Kratica J, Fijuljanin J. 2011. Hybrid genetic algorithm for solving of maximum betweenness problem. Proceedings of the 1st International and 10thBalkan Conference on Operational Research –BALCOR. p. 402-408.

Seffert U, Schleifer FM, Zuhlke D. 2011. Recent Trends in Computational Intelligence in Life Sciences. Proceedings of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning–ESANN [accessed 25 Apr 2017]; https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2011-5.pdf

Slonim D, Stein D, Kruglyak L,Lander E. 1997. Building human genome maps with radiation hybrids. Journal of Computational Biology. 4(4): 487-504.

Su CT, Lin HC. 2011. Applying electromagnetism-like mechanism for feature selection. Information Sciences, 181(5): 972-986.

Tavakkoli-Moghaddam R, Khalili M, Naderi B. 2009. A hybridization of simulated annealing and electromagnetic-like mechanism for job shop problems with machine availability and sequencedependent setup times to minimize total weighted tardiness. Soft Computing, 13(10): 995-1006.

Vapnik V. 1999. An overview of statistical learning theory. IEEE Transactions on Neural Networks.10(5):988-999 .

Varlamis I, Apostolakis I, Sifaki-Pistolla D, Dey N, Georgoulias V, Lionis C, 2017. Application of data mining techniques and data analysis methods to measure cancer morbidity and mortality data in a regional cancer registry: The case of the island of Crete, Greece. Computer Methods and Programs in Biomedicine. 145:73-83.

Verma B, Zhang P. 2007. A novel neural-genetic algorithm to find the most significant combination of features in digital mammograms. Applied Soft Computing. 7(2): 612-625.

Vieiraa S, Mendonca L, Farinhaa G, João S. 2013. Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. Applied Soft Computing. 13(8): 3494-3504.

Yang J, Honavar V. 1998. Feature subset selection using a genetic algorithm. IEEE Intelligent Systems and their Applications, 13(2): 44-49.

Yan K,Xu Y, Fang X, Zheng C , Liu B. 2017. Protein fold recognition based on sparse representation based classification. Artificial Intelligence in Medicine [accessed 25 Apr 2017]; http://www.aiimjournal.com/article/S0933-3657(16)30600-5/fulltext. doi: http://dx.doi.org/10.1016/j.artmed.2017.03.006.

Zhang X, Chen X, He Z. 2010. An ACO-based algorithm for parameter optimization of support vector machines. Expert Systems with Appl, 37(9): 6618-6628.