Soft Comput
DOI 10.1007/s00500-013-1180-x

METHODOLOGIES AND APPLICATION

Electromagnetism-like algorithm for support vector machine parameter tuning

Aleksandar Kartelj · Nenad Mitić · Vladimir Filipović · Dušan Tošić

Abstract This paper introduces an electromagnetism-like (EM) approach for solving the problem of parameter tuning in the support vector machine (SVM). The proposed method is used to tune binary SVM classifiers in single and multiple kernel mode. The internal kernel structure is based on linear and radial basis functions (RBF). An appropriate encoding scheme of EM enables easy transformation of real-valued EM points directly to real-valued parameter combinations. Estimations of the generalization error based on the cross-validation and validation set error are used as objective functions. The efficient local search procedure uses variable size interval movement in order to improve the convergence of the method. The quality of the proposed method is tested on four collections of testing benchmarks through five separate experiments. The first three collections consist of small-size to medium-size classification data sets with up to 60 features and 1,300 training vectors, while the fourth collection is formed of large heterogeneous data sets with up to 1,554 features and 2,186 training vectors. The obtained results indicate that EM outperforms the comparison algorithms in 10 out of 13 instances from the first collection, 5 out of 5 instances from the second, and 13 out of 15 instances from the third collection. The last two experiments, conducted on the fourth collection, show that the proposed method outperforms 14 successful methods in 3 out of 5 data sets where RBF multiple kernel learning is used, and behaves competitively in cases when linear kernels are used.

Keywords SVM parameter tuning · Electromagnetism-like metaheuristic · Classification

Communicated by V. Piuri.

A. Kartelj (✉) · N. Mitić · V. Filipović · D. Tošić
Faculty of Mathematics, University of Belgrade,
Studentski Trg 16, Belgrade 11000, Serbia
e-mail: kartelj@matf.bg.ac.rs

N. Mitić
e-mail: nenad@matf.bg.ac.rs

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

D. Tošić
e-mail: dtosic@matf.bg.ac.rs

1 Introduction

Support vector machine (SVM) is a supervised machine learning technique used for classification and for estimation of functional forms in regression problems, where it is necessary to predict a continuous variable. SVM uses a training data set to build a learning function that generalizes well and produces correct predictions when used on unseen data. 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. In this paper, the binary classification problem is considered. Multiclass classification problems can be reduced to multiple binary classification problems, as presented in Allwein et al. (2001). In the binary classification problem the training data set is composed of feature vectors labelled with one of the two possible classes. The task is to build a hyperplane that separates feature vectors according to their class labels. Additionally, the separating hyperplane should be maximally distant from the vectors on both sides.

The theoretical foundations of SVMs have been defined in Vapnik (1995, 1999). The motivation for employing SVM in classification tasks comes from the result of statistical learning theory where a theoretical upper bound for the generalization error is developed. The upper bound is minimized when the distance between vectors and the separating hyper-

Published online: 29 November 2013

plane is maximized. The important practical property of the bound is its independency from the dimensionality of the feature space. Building a separating hyperplane is not possible when the feature space is not lineary separable. In that case 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. 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 kernel that is defined for each pair of feature vectors. The essential property of the kernel function is that it can be calculated in the original feature space, thus increasing the space dimensionality is unimportant from the perspective of SVM performances. Choosing an appropriate type of kernel (including its inner parameter structure) has high influence on the quality of the SVM prediction ability. Kernel transformation yields a symmetric positive semidefinite matrix that can be regarded as a similarity matrix among training vectors. Conforti and Guido (2010) stress the relevance of selecting the most appropriate kernel function for the corresponding training data since kernel-based methods produce high performances by incorporating prior knowledge through kernel function. In Conforti and Guido (2010), the authors obtain suitable kernel matrices by forming linear combinations of known kernel matrices using a semidefinite programming approach. After that, they successfully apply the combined kernel matrices within the SVM to perform medical diagnostic decision making, i.e. classification tasks. 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 parameters making an impact on the overall prediction quality. Gascón-Moreno et al. (2013) made an overview of several multiple kernel methods within the SVM framework, and concluded, among other things, that combining kernels can be more useful than using a single kernel.

In this paper, we consider the problem of parameter determination in the SVM learning process. Tuning the SVM parameters is a difficult task since there can be many of them and their values usually belong to a large domain of real values. The standard approach for establishing their values is called grid search (GS). It is based on a discretization of real-valued domain to a grid of values, after which an exhaustive exploration over the whole discretized domain is made. This means that all possible combinations of parameter values are checked. GS is computationaly an expensive method and it usually fails to produce good results when the number of parameters exceeds 2. Therefore, heuristic approaches that do not perform such an exhaustive exploration are prefferable.

The subsequent portion of the present section is concerned with the following matter: in Sect. 1.1, we present the definition of the classification problem, and the way it is being formulated and solved by the SVM; after that, single and multiple kernel learning (MKL) is discussed in more detail. The second section of the paper contains a literature review of the methodologies that deal with the problem of SVM parameter tuning. In the third section, the proposed algorithm is described through several subsections, where each describes one of the algorithms essential aspects, i.e. objective function calculation, local search (LS) etc. Our findings, based on the exhaustive experimentation with different data collections are presented in the fourth section. Here, we also provide the evidence for the quality of the algorithm by conducting a statistical analysis. The final, fifth section, holds the conclusions and guidelines for further research.

1.1 Problem definition

Definition of a binary classification problem and the way it is adopted to SVM (Kecman 2001; Phienthrakul and Kijsirikul 2010; Campbell and Ying 2011), as well as the single and MKL models, are presented in this section.

Let us consider the binary classification problem where D_{tr} denotes training data set, composed of N_{tr} pairs of form $(\mathbf{x_i}, y_i), i = 1, \ldots, N_{tr}$, where $\mathbf{x_i} \in \mathbf{R}^N$ is N-dimensional feature vector and $y_i \in \{-1, 1\}$ is a corresponding class label. SVM employs training data set D_{tr} in order to find a separating hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$, $\mathbf{w} \in \mathbf{R}^N$, $b \in \mathbf{R}$ that is maximally distant from the training vectors on each side. The separating hyperplane is then easily transformed to prediction (decision) function $f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$ that calculates the class label for a given unseen vector. Strictly separable data sets are very rare in practice, therefore, a slight modification of the above formulation is proposed by Cortes and Vapnik (1995). Instead of having a strict margin between positively and negatively classified examples, the so-called *soft margin*: $y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 - \zeta_i$, $i = 1, \ldots, N_{tr}$ allows data vectors to be on the wrong side, but close to separating boundary. ζ_i is non-negative slack variable representing the overlapping error related to the training vector $\mathbf{x_i}$. Based on this modified definition of separating hyperplane, the optimal values for $\mathbf{w}$ and b can be found by solving the following optimization problem:

$$\min\left(\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N_{tr}}\zeta_i\right) \tag{1}$$

subject to:

$$y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 - \zeta_i, \; i = 1, \ldots, N_{tr} \tag{2}$$

$$\zeta_i \geq 0, \ i = 1, \ldots, N_{tr} \tag{3}$$

C is the regularization (penalty) parameter that controls the influence of overlapping errors.

SVM uses a convinient dual representation for the assessment of the separating hyperplane. Without going into explicit derivation, a corresponding SVM dual (Boser et al. 1992) is given as follows:

$$\max \left(\sum_{i=1}^{N_{tr}} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N_{tr}} y_i y_j \alpha_i \alpha_j \mathbf{x_i} \mathbf{x_j} \right) \tag{4}$$

subject to:

$$\alpha_i \in [0, C], \quad i = 1, \ldots, N_{tr} \tag{5}$$

$$\sum_{i=1}^{N_{tr}} \alpha_i y_i = 0 \tag{6}$$

where $\alpha_i, \ i = 1, \ldots, N_{tr}$ denote Lagrangian multipliers, while C in this formulation represents their upper bound. Conseqently, C controls the overall maximization expression, i.e. the tradeoff between classifier margin maximization and error minimization.

In the case when data are not lineary separable, the original feature space should be transformed. This is done by replacing each vector $\mathbf{x_i}$ with $\Phi(\mathbf{x_i})$, where $\Phi : \mathbf{R}^N \rightarrow \mathbf{R}^{N'}$ maps the original space to the one which allows the linear separation of data. The exact form of Φ is not relevant as long as the inner product between two vectors in the mapped space $K(\mathbf{x_i}, \mathbf{x_j}) = \Phi(\mathbf{x_i}) \cdot \Phi(\mathbf{x_j})$ is known. $K : \mathbf{R}^N \times \mathbf{R}^N \rightarrow \mathbf{R}$ is called SVM kernel and represents a similarity (or distance) metric between the input feature vectors $\mathbf{x_i}$ and $\mathbf{x_j}$. In order to be applicable, kernel has to satisfy Mercers's condition (Shawe-Taylor and Cristianini 2004). After the mapping of the feature space, the dual maximization term becomes:

$$\max \left(\sum_{i=1}^{N_{tr}} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N_{tr}} y_i y_j \alpha_i \alpha_j K(\mathbf{x_i}, \mathbf{x_j}) \right) \tag{7}$$

Finally, its asociated decision function becomes:

$$f(\mathbf{x}) = \operatorname{sgn} \left(\sum_{i=1}^{N_{tr}} \alpha_i y_i K(\mathbf{x_i}, \mathbf{x}) + b \right) \tag{8}$$

It is shown that the value of the parameter C and the type of the kernel function are tightly related to the overall classification error of SVM (Lavesson and Davidsson 2006). Appropriate value of the parameter C is usually determined from a large positive domain of real numbers. Additionally, allowing the usage of different kernel functions, that can also be parameterized, makes the problem even more challenging. In the following two subsections, kernel learning models and their parameter structures are discussed.

1.2 Single kernel learning (SKL)

Single kernel learning deals with tuning of only one set of parameters, i.e. parameters for only one kernel. This technique is usually preferable when data features do not form clusters (are not group-related). As previously mentioned, there is a regularization parameter C and kernel function K which can take various forms. In our research two single kernel models are considered: linear (9) and radial basis (10).

$$K(\mathbf{u}, \mathbf{v}) = \mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^{N} u_i v_i \tag{9}$$

$$K^{\sigma}(\mathbf{u}, \mathbf{v}) = \exp \left(- \sum_{i=1}^{N} \frac{(u_i - v_i)^2}{2\sigma_i^2} \right) \tag{10}$$

The linear kernel model 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. In the latter case, the kernel function itself is parameterized, so the parameter set to be tuned is $\{C, \sigma_1, \sigma_2, \ldots, \sigma_N\}$, where σ_i is called scaling factor, and it corresponds to a radius of radial basis function used for the ith feature. The scaling factors can be useful for real-world databases containing many features of different nature (Chapelle et al. 2002). In our paper, a relaxed version of the parameter space for SKL is used by setting $\sigma_1 = \sigma_2 = \cdots = \sigma_N = \sigma$. Consequently, the problem is reduced to a search over the $\{C, \sigma\}$ set of parameters.

1.3 Multiple kernel learning

Multiple kernel learning (MKL) uses more complex problem formulation than SKL models. In these models the features are distributed over N_G clusters, which represent non-overlapping groups of features separated according to their different natures. G_j denotes the j-th subset of features. This leads to the introduction of multiple kernels $K_1, \ldots, K_{N_G}$ with different parameters. The kernels can be later aggregated into linear, conic, convex combinations, or some other nonlinear functional form that uses multiplication, power function, exponentiation, etc. Gascón-Moreno et al. (2013) made a detailed overview of different MKL models. In our research the conic combination of linear (11) and radial basis function kernels (12) is used.

$$\begin{aligned} K^{\alpha}(\mathbf{u}, \mathbf{v}) &= \sum_{j=1}^{N_G} \alpha_j K_j(\mathbf{u}, \mathbf{v}) \\ &= \sum_{j=1}^{N_G} \alpha_j \sum_{i \in G_j} u_i v_i, \end{aligned} \tag{11}$$

$$\alpha_j > 0, \ j = 1, \ldots, N_G$$

$$K^{\alpha,\sigma}(\mathbf{u},\mathbf{v}) = \sum_{j=1}^{N_G} \alpha_j K_j^{\sigma_j}(\mathbf{u},\mathbf{v}) = \sum_{j=1}^{N_G} \alpha_j \exp\left(-\sum_{i\in G_j} \frac{(u_i - v_i)^2}{2\sigma_j^2}\right), \quad (12)$$

$\alpha_j > 0,\ j = 1,\ldots,N_G$

It is clear that MKL deals with higher dimensional parameter space than SKL. This is due to the introduction of coefficients $\alpha_j, j = 1,\ldots,N_G$ in the conic combination, and scaling factors $\sigma_j, j = 1,\ldots,N_G$ in the case of radial basis function kernels. Therefore, the total number of parameters to be tuned, denoted by H, is N_G for the linear model, and $2N_G$ for the radial basis model (this also holds for SKL models where $N_G = 1$).

2 Related work

Different search strategies, i.e. the ways how the parameter region can be efficiently traversed are proposed in the literature.

Keerthi and Lin (2003) proposed an improved grid search of parameters for SKL model that exploits asymptotic behavior of radial basis function kernels. Keerthi (2002) used a radius/margin bound as a performance measure allowing *tackling* of high dimensional problems with more than 10,000 support vectors. Keerthi (2002) employed a smoothed k-fold cross-validation and its gradient with respect to SVM parameters to obtain near-optimal solutions. Sequential minimal optimization (SMO) technique to solve quadratically constrained quadratic program (QCQP) is used by Bach et al. (1996). This technique corresponds to the optimization of the coefficients in the conic combination of multiple kernels in SVM. Sonnenburg et al. (2006) showed that the QCQP can be rewriten as a semi-infinite linear program and efficiently solved by using the standard SVM implementations.

Friedrichs and Igel (2005) proposed a method that uses covariance matrix adaptation evolution strategy (CMA-ES). This approach is applicable for high dimensional parameter spaces and produces better results than greed search on smaller instances. Barbero et al. (2009) present two focused grid search (FGS) algorithms. The first variant, deterministic FGS (DFGS), makes a systematic search and performs much faster than standard grid search. The second, annealed FGS (AFGS) introduces elements of randomness which reduces the computational cost. Both algorithms are shown to be competitive to CMA-ES, and easy for usage because of their parameter-free nature.

Several metaheuristic approaches can be found in the literature. Imbault and Lebart (2004) proposed genetic algorithm and simulated annealing approach for parameter tuning. Both algorithms performed robustly and achieved near-optimal solutions. The genetic algorithm is faster, but requires more control parameters to be set. Samadzadegan et al. (2010) proposed a genetic algorithm for SVM parameter tuning that outperforms traditional grid search approach in regards to classification accuracy. Phienthrakul and Kijsirikul (2010) used evolutionary strategies for adjusting the parameters of SVM. Additionally, multi-scale radial basis function kernels (RBF) are weighted and combined producing better discrimination in the feature space. Zhang et al. (2010) applied ant colony optimization for SVM parameter tuning. The algorithm is shown to be succesfull in solving five real-life benchmarks from UCI Machine Learning Repository (Frank and Asuncion 2010). Multi-objective artificial immune algorithm for tuning the kernel, and regularization parameters of SVM, is proposed by Aydin et al. (2011). The algorithm is also succesfully applied to fault diagnosis of induction motors and anomaly detection problems. Gascón-Moreno et al. (2011) described a multi-parametric kernel Support Vector Machine Regression (SVMr) algorithm optimized with an evolutionary technique. This algorithm is well suited for the forecasting problems. The authors also calculated new bounds for the multi-parametric kernel that reduces the SVMr hyper-parameters search space. Similarly, in papers by Zhiqiang et al. (2013) and Gascón-Moreno et al. (2013), SVM is being optimized by a metaheuristic algorithm, and later applied for forecasting. Zhiqiang et al. (2013) use particle swarm optimization to optimize the SVM, and apply it in the prediction of stock price movement on the market, while Gascón-Moreno et al. (2013) use evolutionary algorithm to tune multi-parametric structure of the SVM and later utilize it for different real regression problems from public repositories. Recently, variable neighboorhod search (VNS) (Carrizosa et al. 2012) is adopted for solving this problem. The authors performed several experiments on single and multiple kernel models. The testing results were compared to other state-of-the-art approaches reviewed in Gascón-Moreno et al. (2013).

3 The proposed EM algorithm

Electromagnetism-like algorithm (EM), proposed by Birbil and Fang (2003), represents a population-based optimization technique inspired by mechanisms of interaction among electrically charged particles (points). The method employs a proficient search process governed by, so called, EM points where each one represents single candidate solution of the underlying problem. EM points encoding better solutions are awarded with higher charge. This is crucial for leading the search process towards promising solution regions, because EM points with higher charge attract other points more strongly. The exact attraction-repulsion relationship is given

through Coulomb's Law. Birbil et al. (2004) address consideration about the convergence of EM. Electromagnetism-like algorithms turn out to be successful in solving many problems with practical and theoretical background: EM method for constrained global optimization (Ali andGolalikhani 2010); Su and Lin (2011) adopt EM technique to solve feature selection problem; Tavakkoli-Moghaddam et al. (2009) proposed a hybrid algorithm based on EM and simulated annealing for job shop problem, hybrid EM method for capacitated vehicle routing is proposed by Yurtkuran (2010), and for uncapacitated multiple allocation hub location problem by Filipović (2011). Cuevas et al. (2012) proposed EM algorithm for solving an automatic detection of circular shapes in noisy images. Beside these, several other problems are successfully tackled by EM: unicost set covering problem (Naji-Azimi et al. 2010); strong minimum energy topology problem (Kartelj 2012); maximum betweenness problem (Filipović et al. 2013), etc.

The main elements of the proposed algorithm for parameter tuning are outlined in Fig. 1. 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 through procedure $createInitialPoints$ and after that, the algorithm enters into the main loop. The main loop iterates N_{it} times and in each iteration every EM point is subjected to the objective value calculation, i.e. measuring the quality of solution represented by that point. Upon finishing of the inner loop, local search procedure is applied to at most one of the solution points. The mechanism of choosing that point is described in the Sect. 3.3. The next step is the calculation of the EM points charges. As previously mentioned, the charge of a fixed EM point will depict its solution quality. Charges are then used for calculating the resulting force vectors for each point. EM point movement is guided by direction and magnitude of corresponding force vector. The phases of the EM algorithm are described in the following sections, emphasizing the crucial aspects of the underlying parameter tuning problem. In addition, we provide the visual representation of the proposed methodology via flow chart (Fig. 2).

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

Fig. 1 The proposed EM method

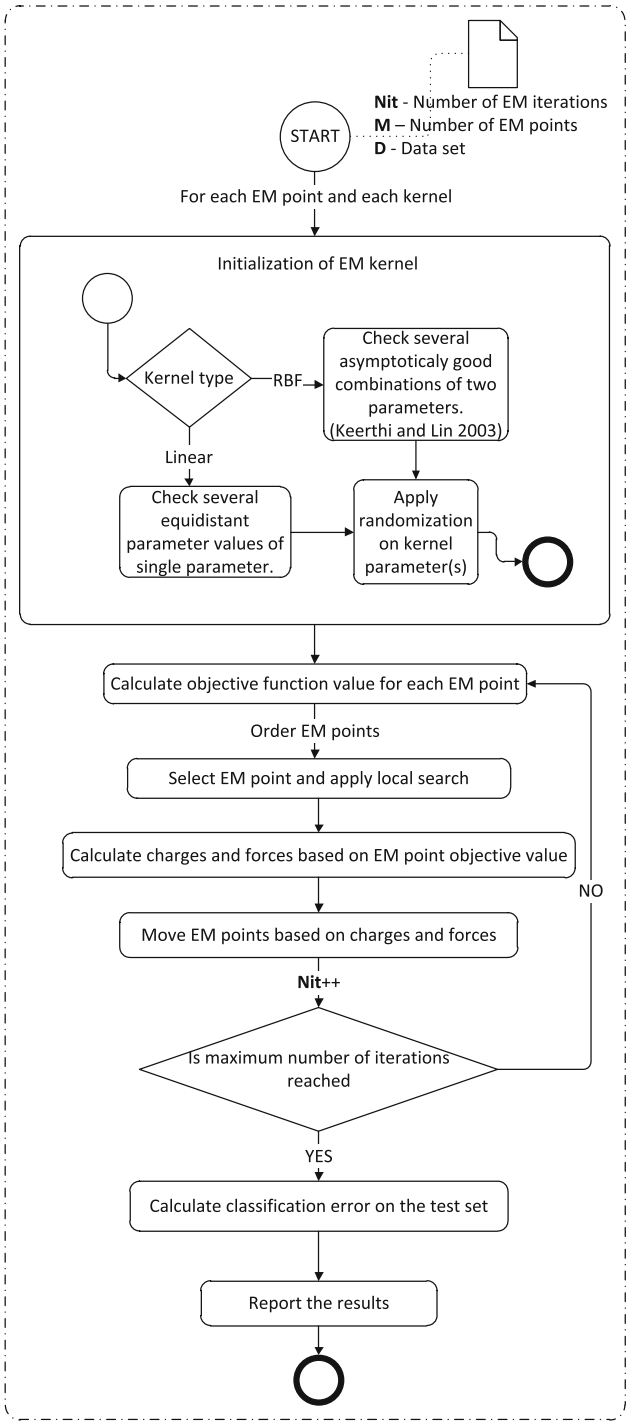

Fig. 2 The process chart of the proposed method

3.1 Initialization

In the initialization phase, a population $\mathbf{p}$ of M initial solutions is established, i.e. the initial values of the kernel

parameters for each of EM points are set. Each EM point is H-dimensional vector of real values. In the radial basis SKL model, i-th EM point encoding is given by: $(p_i^1, p_i^2) \rightarrow (C, \sigma)$. In the linear MKL model i-th EM point encodes the parameters as follows: $(p_i^1, \ldots, p_i^{H=N_G}) \rightarrow (\alpha_1, \ldots, \alpha_{N_G})$, while in the RBF MKL model coefficients from the conic combination and scaling factors are interleaved: $(p_i^1, \ldots, p_i^{H=2N_G}) \rightarrow (\alpha_1, \sigma_1, \ldots, \alpha_{N_G}, \sigma_{N_G})$.

The procedure for establishing the initial values consists of two steps. In the first step, the deterministic heuristic (Keerthi and Lin 2003) is used. The second step applies the randomization procedure to the results obtained in the first step. Throughout this paper, logarithmic notation for representing the parameters is used, because it is more convenient when dealing with large real domains.

Keerthi and Lin (2003) make an analysis of asymptotic behavior of radial basis function kernels, i.e. they discuss how parameters C and σ^2 interact when one of them converges to infinity. They also define the boundary area for *good* solution region as:

$$\log \sigma^2 = \log C - \log \widetilde{C} \tag{13}$$

Here, for any fixed value of $\log \widetilde{C}$ there exists a linear relationship with unit slope between $\log \sigma^2$ and $\log C$. The authors show that when $\sigma^2 \rightarrow \infty$ along that line, the SVM classifier converges to linear classifier with penalty parameter value of $\log \widetilde{C}$. Based on the introduced relationship, a simple heuristic for searching the space of possible values of two parameters is proposed. The technique from Keerthi and Lin (2003) is adopted as a baseline for our initialization procedure. Due to the fact that this technique produces only one solution, we incorporate the randomization step to produce the population of different initial solutions (different EM points).

According to Keerthi and Lin (2003), the $\log \widetilde{C}$ is determined as the best approximation for parameter $\log C$ of linear SVM classifier. In the same way, we calculate the classification error over six equidistant values of $\log C$ in interval $[-8, 2]$. The $\log C$ value that produces the least classification error is denoted as $\log \widetilde{C}$.

Values are assigned to EM points in the folowing way:

- In the case when linear classifier is used, the i-th EM point is initialized as follows: $p_i^j = \log \widetilde{C} X_j, \ j = 1, \ldots, H$, where $X_j \sim \mathcal{N}(1, 1)$.
- In the case when radial basis function kernel is used, the combination of parameters $(\log C, \log \sigma^2)$ offering the highest classification accuracy is sought. This is done by varying values of $\log \sigma^2$ on a grid of six equidistant values from region $[-8, 8]$. Value of $\log C$ is calculated based on (13).

  The combination of parameters that produces the highest classification accuracy from six pairs is denoted as $(\overline{\log C}, \overline{\log \sigma^2})$. Finally, the i-th EM point is initialized as follows: $p_i^{2j} = \log \widetilde{C} + \overline{\log \sigma^2} X_j$, $p_i^{2j+1} = \overline{\log \sigma^2} X_j$, $j = 1, \ldots, N_G$. As before, X_j is random variable selected from $\mathcal{N}(1, 1)$ distribution.

In order to simplify the application of EM procedures, the initial values of parameters are scaled to [0, 1]. Later, in the calculation of the objective function, the values are scaled back to the original interval and used to train the SVM.

3.2 Objective function

In order to sucessfully tune the SVM parameters, EM algorithm is guided by a specific objective function. This function reflects the quality of the solution represented by the EM point. The natural choice for the objective function is the estimation of generalization error of the SVM classifier. The straightforward estimation of the generalization error is the classification error on the training set. Beside this approach, other, more subtle measures are proposed in literature. For example, leave-one-out (LOO) estimation is performed by removing each training vector from the training set, building a classifier, and then testing on the removed vector. The overall estimation of the classification error is finally calculated as a sum of errors for all removed training vectors. It is known that LOO gives almost unbiased estimation of the expected generalization error (Luntz and Brailovsky 1969). Chapelle et al. (2002) compared different variations of LOO. Although LOO seems to be a reasonable measure, it is not very efficient because it requires a large number of SVM executions (one for every training sample). Relaxation of LOO is k-fold cross-validation-based estimation of the expected generalization error. It is calculated by splitting the training set into k folds, and afterward using each fold as a validation set, while the remaining folds are used for 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. More detailed insight about SVM generalization performance estimation is made by Joachims (2000), while Duan et al. (2003) present an overview of simple evaluation measures for parameter tuning.

Figure 3 shows the outline of the procedure for the objective function calculation. The first two steps of the procedure 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 in all computational experiments, with exception in the comparison to ant colony optimization algorithm for SVM tuning. Here, instead of using cross-validation technique, the objective function is calculated as the true error on a validation set [see Zhang et al. (2010) and Chapelle et al. (2002)]. For small problem instances with homogenous (non clustered) features, fivefold cross-

```
input: p_i, D_tr
r = decodeToParameters(p_i);
precomputeKernels(r, D_tr);
p_i^obj = 0;
if not crossValidation then
    p_i^obj = validationSetError();
    return;
end
if N_G == 1 then
    folds = 5;
    p_i^obj = cvEstimate(folds, D_tr);
else
    folds=2;
    for k ← 1 to 5 do
        p_i^obj=p_i^obj+cvEstimate(folds, D_tr);
    end
    p_i^obj = obj/5;
end
```

Fig. 3 Objective value calculation

validation is employed, and for the large instances the classification error estimation is based on the 5 × 2-fold cross validation. In the first case, the training set is divided into 5 equally sized non-overlapping subsets, and then in 5 separate iterations each of these subsets is used for measuring classification error while the others are used for training the classifier. At the end, the averaged classification error on these 5 subsets is used as a classification error estimation. In the second case, twofold cross validation is performed 5 times, and average error across these 5 runs is recorded.

3.3 Local search

Local search procedure is separated into two phases (Fig. 4): the first is the selection of the point on which the LS is going to be applied, and the second is applying of LS.

The selection phase is similar to one described in Filipović et al. (2013). Candidates for LS are the best and the second best point. Therefore, EM points are first sorted in ascending order with respect to the classification error estimation. Procedure *applicableLS* checks if the EM point fulfill necessary requirements, i.e. whether the LS has never been applied to this point before, or it has been applied, but the value has changed since the last applying of LS. In case the best point does not meet this requirement, the second best point is checked. If LS is not applicable at all, the procedure finishes. The motivation for using these criteria for performing LS is twofold:

1. Firstly, the number of calls for evaluation of the objective function is considerably reduced, implying significant reduction of execution time.
2. Secondly, performing LS on each point could decrease exploratory properties of the search algorithm.

```
input: p, D_tr
sortByClassificationErrorAsc(p);
ind = -1;
if applicableLS(p_1) then
    ind = 1;
else if applicableLS(p_2) then
    ind = 2;
end
if ind == -1 then
    return;
end
for k ← 1 to H do
    for sign ← 0 to 1 do
        step = (sign-p_ind^k)/10;
        improved = true;
        while improved = true do
            improved=false;
            oldObj=p_ind^obj;
            oldCoord=p_ind^k;
            p_ind^k = p_ind^k + step;
            newObj=objectiveFunction(p_ind, D_tr);
            if newObj < oldObj then
                improved = true;
            else
                p_ind^k = oldCoord;
                p_ind^obj = oldObj;
            end
        end
    end
end
```

Fig. 4 Local search

The second phase of the procedure is searching for an improvement for the actual parameter setting. The algorithm attempts to find an improvement in both directions for every coordinate of the selected EM point: towards left (value 0) and right boundary value (value 1). This is done by increasing the value of coordinate by 1/10 of the remaining interval on left and right side of the current coordinate value. When improvement is found, it is immediately applied and the search for new improvement is continued in the same direction. If improvement is not found, the search process continues in the opposite direction once. After all encoded kernel parameters are checked for improvement, the algorithm ends.

3.4 Charges and forces

Objective values of EM points indirectly, through charges and forces, define the movement through solution space regions. The calculation of charge takes into account the objective values of whole EM population (14).

$$q_i = exp\left(-H\frac{p_i^{obj} - p_{best}^{obj}}{\sum_{k=1}^{M}(p_k^{obj} - p_{best}^{obj})}\right) \tag{14}$$

The intuition behind mapping better EM point objective values (lower objective value is better) to higher charges is according to the fact that the better EM points should have more important role in leading the search process. This is accomplished here by assigning them higher attraction-repulsion property. In (14), q_i denotes the charge of i-th EM point. It can be seen that the point with the best objective value gets charge of 1, while the others get some value from $(0, 1]$.

After the calculation of charges, the forces of interaction between each pair of points are calculated. For a given EM point $\mathbf{p_i}$, $\mathbf{F_i}$ is total force that influences this point and its calculation is shown in (15).

$$\mathbf{F_i} = \begin{cases} \sum_{j=1, j\neq i}^{M}(\mathbf{p_j} - \mathbf{p_i})\frac{q_i \times q_j}{||\mathbf{p_j} - \mathbf{p_i}||^2}, & p_j^{obj} < p_i^{obj} \\ \sum_{j=1, j\neq i}^{M}(\mathbf{p_i} - \mathbf{p_j})\frac{q_i \times q_j}{||\mathbf{p_j} - \mathbf{p_i}||^2}, & p_j^{obj} \geq p_i^{obj} \end{cases} \tag{15}$$

As it can be seen, the force $\mathbf{F_i}$ is obtained by super-positioning all pairwise interactions between $\mathbf{p_i}$ and other EM points. Single parwise interaction between points $\mathbf{p_i}$ and $\mathbf{p_j}$ is defined by expression similar to Coulomb's Law, i.e. the interaction is proportional to charges, and inversely proportional to distances. As a final step, movement procedure is applied. The direction and magnitude of the movement is led by previously calculated forces (16).

$$p_i^k = \begin{cases} p_i^k + \lambda\frac{F_i^k}{||\mathbf{F_i}||}(1 - p_i^k), & F_i^k > 0 \\ p_i^k + \lambda\frac{F_i^k}{||\mathbf{F_i}||} \cdot p_i^k, & F_i^k < 0 \end{cases} \tag{16}$$

Additionally, an effect of randomness is introduced through variable λ. This variable is randomly sampled from $[0, 1]$ in each iteration and for each EM point. In (16), F_i^k denotes the k-th coordinate of force vector corresponding to i-th EM point.

4 Experimental results

In this section the performances of the proposed approach are evaluated through five experiments. The first three are made upon three collections of small-size and medum-size instances, with up to 60 features. The number of parameters H is 2 in both cases. The last two consider utilization of linear and radial basis MKL models on the collection of large instances. As described in Sect. 1.3 the number of parameters is: $H = N_G$ for the linear, and $H = 2N_G$ for the radial basis kernel model.

Table 1 Small instances used in the first and partially in the second experiment

inst.	N_{tr}	N_{ts}	N
Banana	400	4,900	2
Diabetes	468	300	8
Image	1,300	1,010	18
Splice	1,000	2,175	60
Ringnorm	400	7,000	20
Twonorm	400	7,000	20
Waveform	400	4,600	21
German	700	300	20
Heart	170	100	13
Thyroid	140	75	5
Titanic	150	2,051	3
Solar	666	400	9
Breast cancer	200	77	9

Experiment 1 Table 1 contains information about 13 instances (denoted as *the first experimental collection* in further text) on which the comparison with Keerthi and Lin (2003) and Carrizosa et al. (2012) is based. It consists of the following columns: instance name ($inst.$), number of training samples (N_{tr}), number of testing samples (N_{ts}) and number of features (N). The instances are available through the Machine Learning UCI repository (Frank and Asuncion 2010). However, we used an alternative, purified collection of the same instances, provided by Rätsch et al. (2001), who made a partitioning of each of these instances to training and testing set in 100 different ways. As in Keerthi and Lin (2003) and Carrizosa et al. (2012), we use only the first of these 100 partitions for each instance.[1]

We refer the reader to the seminal paper of Muller et al. (2001), in which the first experimental collection was used to compare several kernel-based classification techniques, namely: SVM, kernel fisher discriminant analysis (KFD), radial basis function (RBF), AdaBoost (AB) and regularized AdaBoost (AB_R). Neither of these methods showed to be systematically best, which is expected, since all algorithms have similar kernel-based foundations. However, KFD and SVM produced results of higher quality. In later research conducted by Franc and Hlaváč (2003), it was shown that SVM performs better than KFD on the subset of this collection. A vast number of researches, whose review is out of the scope of this paper, point out to a conclusion that SVM often outperforms the compared classification algorithms (even in its pure form, without parameter optimization) when applied to instances from this collection. The superiority of SVM classifier, confirmed by those researches, supports our decision

[1] Data sets available at: http://mldata.org/repository/tags/data/IDA_Benchmark_Repository/.

to compare the proposed algorithm only to other parameter tuned SVM classifiers.

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

Experiment 2 The second experiment is based on the collection, which is a subset of the first collection, since it considers only the following datasets: *Breast Cancer, Diabetes, Heart, Thyroid* and *Titanic*. These five instances are used in order to compare EM to the ACO technique for SVM tuning proposed by Zhang et al. (2010). As in the first experimental collection, the first of one hundred dataset partitioning, according to Rätsch et al. (2001), is used. The way the objective function is calculated differs from the previous experiment, since here, the validation set error is used.

Experiment 3 The third collection of small instances is the one used by Phienthrakul and Kijsirikul (2010). These data sets are aquired directly from the authors. Each of the 15 instances is divided five different ways into the training and testing part (five folds where each one is used once as a testing part). As we can see from Table 2, dimensions are quite similar to those used in the first experiment, i.e. in the first benchmark collection. The validation and testing methodology is the same as in Phienthrakul and Kijsirikul (2010), i.e.

Table 2 Small instances used in the third experiment

inst.	N_{tr}	N_{ts}	N
Checkers	153	39	2
Spiral	465	117	2
Liver disorders	276	69	6
Indians diabetes	614	154	8
Three of nine	410	102	9
Tic tac toe	766	192	9
Breast cancer	559	140	10
Parity bits	819	205	10
Solar flare	853	213	10
Cleveland heart	216	54	13
Australian	552	138	14
German-org	800	200	24
Ionosphere	280	71	34
Tokyo	767	192	44
Sonar	166	42	60

Table 3 Large instances used in the third and the fourth experiments

inst.	N_{tr}	N_{ts}	N	G
mfEO4	1,333	667	427	(76, 64, 240, 47)
mfEO6	1,333	667	649	(76, 64, 240, 47, 216, 6)
mfSL4	1,333	667	427	(76, 64, 240, 47)
mfSL6	1,333	667	649	(76, 64, 240, 47, 216, 6)
adv	2,186	1,093	1,554	(457, 495, 472, 111, 19)

the fivefold cross validation on the training set is used as an objective function, while the prediction model is built upon the best EM point. The model is later used to perform evaluation of the testing subset, and the average of test errors accross 5 different partitionings of training and testing set is recorded for comparison.

Experiments 4 and 5 For the fourth and the fifth experiments, 5 large instances[2] as well as the experimental methodology desribed by Carrizosa et al. (2012) and Gascón-Moreno et al. (2013) are adopted. Contrary to the small instances, here the features are heterogeneous, i.e. they are grouped into the clusters and that grouping is known in advance. The last column of Table 3 shows cardinalities of groups (G). In the fourth and fifth experiment, linear and radial basis MKL models are employed. Each kernel in these models deals with a single feature cluster (grouping). Both linear and radial basis kernel models are evaluated by using classification error estimation that is based on 5×2-fold cross validation. For previosly described large instances, partitioning of starting data set to training and testing subsets is not provided, therefore, as in Gascón-Moreno et al. (2013) and Carrizosa et al. (2012), the training and testing partitions are randomly sampled in ratio 2:1. Average testing error across multiple executions that use different random seeds is later used for the purpose of comparison.

Experimental environment should simultaneously keep uniform control parameters and replicate similar execution conditions throughout all compared algorithms. More specifically, it is extremely important to use approximatelly the same number of objective function evaluations when comparing different algorithms (Črepinšek et al. 2012). In some cases, when there are no additional exploration/exploitation steps (e.g., local search) the algorithms can be compared under the same number of iterations. Here, it is not the case, since the proposed EM algorithm contains local search.

Therefore, the first set of instances was solved by using $M = 5$, and $N_{it} = 5$. These control parameters are set with the intention to achieve approximately the same number of objective function calculations (54) as it was the case in

[2] Data sets available at: http://archive.ics.uci.edu/ml/datasets/Multiple+Features, http://archive.ics.uci.edu/ml/datasets/Internet+Advertisements.

Keerthi and Lin (2003) and Carrizosa et al. (2012). This number of calculations also includes objective function calculations from LS. The other two experiments on small instances used far more objective function calculations. Zhang et al. (2010) use the population size of 80 ants. Therefore, we set $M = 80$ in the second experiment. However, ACO finishing criterion is not based on the maximal number of iterations, and the total number of evaluations is not reported, therefore we set $N_{it} = 1{,}000$ and show computational times for each of the compared algorithms. Phienthrakul and Kijsirikul (2010) used a large number of objective function calculations in their evolutionary strategy (ES). Their ES iterated for 1,000 times, and in each iteration it calculated the objective value of 10 candidate solutions, giving a total of 10,000 calculations. Therefore, the setup for this experiment was: $M = 100$, $N_{it} = 100$. The actual number of evaluations in our replicated environment was usually a few hundred evaluations higher than in Phienthrakul and Kijsirikul (2010) because of LS. For the large scale instances, M is set to 8 and N_{it} to 15. Obtained results indicate that the number of objective value evaluations for those parameters oscillates around 600, which is the total number of iterations, i.e. the objective function calculations in Carrizosa et al. (2012).

The proposed EM algorithm is written in C programming language and compiled with Visual Studio 2010 compiler. All tests were carried out on the Intel Xeon E5410 @ 2.34 GHz.

Table 4 contains the results from the first experiment, i.e. the comparison among the proposed method and two methods from literature. Columns denoted by KL and VNS refer to the results obtained by Keerthi and Lin (2003) and Carrizosa et al. (2012) after a single execution. The number of objective value evaluations, running time, and the iteration in which the solution was found are also reported in the last three columns of the table. We also include three more columns representing scaled errors for each of the compared algorithms: $\overline{KL}$, $\overline{VNS}$, $\overline{EM}$. These values are later used in the statistical analysis. They are obtained by scaling classification errors of the three compared methods to interval [0,1] for each dataset separately. That way, the worst of the three algorithms for each dataset gets the value of 1, while the best is assigned the value of 0. To prevent division by zero, the scaling is not performed in the case where all algorithms reached the same classification error. In those cases, the scaled error is set to 0 for all algorithms (see Solar instance). The motivation for performing scaling transformation is in the fact that different datasets (instances) vary significantly in terms of magnitudes of classification errors. Therefore, without scaling, certain datasets would have higher relevance in the overall comparison. The results indicate that the proposed method outperforms the other two approaches on 10 out of 13 testing benchmarks, and produces the second best solution on the remaining three instances.

The results from the second experiment are shown in Table 5. The first three columns show the classification errors of SVM tuned by grid search (GS), ant colony optimization (ACO) and the proposed electromagnetism-like algorithm (EM). Both grid search and ant colony optimization results are taken from Zhang et al. (2010). This is, as in the previous experiment, followed by scaled errors. Finally, the corresponding computational times (in seconds) are shown in the last three columns. It is evident that EM outperforms both comparison algorithms, on all tested instances. It can be seen that computational times differ significantly, i.e. EM usually spends less time. This is due to the fact that EM and ACO have different finishing criteria, namely, EM finishes after a given number of iterations, while ACO stops its execution when certain level of precision (called ϵ in Zhang et al. (2010)) is reached.

Table 4 Single RBF kernel on small instances from the first experiment

inst.	KL	VNS	EM	$\overline{KL}$	$\overline{VNS}$	$\overline{EM}$	$Eval_{EM}$	t_{EM}(s)	$Iter_{found}$
Banana	11.59	11.61	11.57	0.48	1	0	57	7.35	1/5
Diabetes	24	24.67	23.33	0.50	1	0	69	9.65	1/5
Image	5.84	2.38	2.18	1	0.06	0	60	34.66	3/5
Splice	10.53	9.93	10.16	1	0	0.38	49	39.34	3/5
Ringnorm	1.44	1.7	1.63	0	1	0.73	61	8.44	1/5
Twonorm	2.47	2.77	2.36	0.27	1	0	53	10.48	2/5
Waveform	11.39	10.46	11.22	1	0	0.81	58	13.6	2/5
German	21.33	21.33	20.33	1	1	0	50	21.88	2/5
Heart	21	20	19	1	0.5	0	35	0.61	1/5
Thyroid	5.33	5.33	4	1	1	0	47	0.52	2/5
Titanic	22.92	22.92	22.57	1	1	0	40	13.82	1/5
Solar	34.5	34.5	34.5	0	0	0	47	11.52	2/5
Breast cancer	29.87	28.57	28.57	1	0	0	41	4.88	2/5

Table 5 Single RBF kernel on small instances from the second experiment

inst.	GS	ACO	EM	$\overline{GS}$	$\overline{ACO}$	$\overline{EM}$	t_{GS}	t_{ACO}	t_{EM}
Breast cancer	25.97	25.97	23.38	1	1	0	2,547.3	1,437.8	270.44
Diabetes	23.33	23	22.67	1	0.5	0	29,078	19,298	1,837.46
Heart	19	16	15	1	0.25	0	1,446.4	519.58	270.71
Thyroid	4	2.67	1.33	1	0.5	0	702.89	666.2	163.43
Titanic	22.57	22.57	21.84	1	1	0	639.95	429.22	1,437.42

Table 6 Single RBF kernel on small instances from the third experiment

inst.	GS	ES	EM	$\overline{GS}$	$\overline{ES}$	$\overline{EM}$	$Eval_{EM}$	t_{EM}(s)	$Iter_{found}$
Checkers	16.68	16.18	43.09	0.02	0	1	10,247.8	53.2	9.2/100
Spiral	0	0	0	0	0	0	11,543.4	603.2	44.4/100
Liver disorders	38.26	33.33	26.96	1	0.56	0	11,510.8	383.47	32/100
Indians diabetes	35.03	26.7	22.4	1	0.34	0	11,413.8	1,749.21	52.4/100
Three of nine	46.49	0	0	1	0	0	11,616.4	495.95	14.8/100
Tic tac toe	34.66	0.31	0	1	0	0	11,335.6	2,475.52	14/100
Breast cancer	13.59	5.44	3.86	1	0.16	0	11,526	663.49	25.2/100
Parity bits	51.95	24.22	51.95	1	0	1	10,232.4	2,598.03	1/100
Solar flare	19.13	19.04	17.26	1	0.95	0	110,861.4	6,173.75	11.4/100
Cleveland heart	44.44	21.85	14.81	1	0.24	0	10,622.8	132.02	9.4/100
Australian	44.49	44.49	28.12	1	1	0	10,747	1,150.56	2.2/100
German-org	29.9	29.7	24	1	0.97	0	10,687.8	2,360.07	5/100
Ionosphere	33.9	4.84	4.84	1	0	0	11,489.4	199.12	22/100
Tokyo	18.98	8.34	7.92	1	0.04	0	11,283	2,182.48	25.4/100
Sonar	29.33	11.08	11.07	1	0	0	11,455	93.29	56.2/100

The results from the third experiment are presented in Table 6. The second and third column reffer to the average classification error of grid search (GS) and ES from Phienthrakul and Kijsirikul (2010). The average results of the proposed EM are showed in the fourth column. As in the first and the second experiment, the scaled errors are shown in the succeeding columns: $\overline{GS}$, $\overline{ES}$ and $\overline{EM}$. The last three columns show the number of the objective function calculations ($Eval_{EM}$), running time (t_{EM}) and the average iteration number when the solution is found ($Iter_{found}$). The results show that the EM algorithm consistently produces the best solutions in 13 out of 15 cases. On one of the remaining instances, *Parity Bits*, EM performs equally good as the grid search algorithm, and only in the case of *Checkers* data set, EM is worse than the other two algorithms. The average number of objective function evaluations is slightly higher than 10,000, but this seems to have no effect on the overall quality, since the solutions are found in less than 33 iterations (out of 100).

The fourth experiment operates with large-scale instances where solutions are obtained by tuning parameters of multiple linear kernel model (11). Averaged results of the proposed algorithm (EM) are compared to best (IGA_b), median (IGA_m), and worst (IGA_w) solution from 12 different models described by Gascón-Moreno et al. (2013).

Table 7 Multiple linear kernel on large instances

inst.	$IGA_{(b,m,w)}$	VNS	EM	$Eval_{EM}$	t_{EM}(s)	$Iter_{found}$
mfEO4	(1.99, 2.15, 4.22)	2.34	2.14	539.7	1,567	3.4/15
mfEO6	(1.61, 1.76, 3.1)	1.46	2.02	654.8	2,076.5	4/15
mfSL4	(4.82, 5.11, 9.46)	6.19	5.56	575.2	1,698	3.7/15
mfSL6	(2.19, 2.54, 10.82)	2.3	5.04	721.5	2,510.6	3.1/15
adv	(3.41, 3.72, 4.9)	3.22	4.76	336.4	4,391.8	2.6/15

Beside that, EM is compared to the nested VNS linear model provided by Carrizosa et al. (2012). Table 7 shows the obtained results, which confirm that the proposed algorithm is competitive, especially on instances mfEO4 and mfSL4.

The final (fifth) experiment is based on the utilization of radial basis function kernels (12). As in the fourth experiment, the solution EM is compared to the best, median and worst solution of 12 algorithms from Gascón-Moreno et al. (2013) that use radial basis kernels. Additionally, a comparison with two nested VNS models VNS_1 and VNS_2, described in Carrizosa et al. (2012), is made. The results showed that the proposed EM algorithm outperforms all other methods on 3 out of 5 instances, while performing competitively on the remaining two (see Table 8).

Table 8 Multiple radial basis kernel on large instances

inst.	$IGA_{b,m,w}$	VNS_1	VNS_2	EM	$Eval_{EM}$	t_{EM}(s)	$Iter_{found}$
mfEO4	(0.67, 0.96, 2.18)	0.74	0.72	0.6	519.8	1,958.6	5.9/15
mfEO6	(0.58, 0.67, 7.22)	0.65	0.53	0.48	633.6	2,525.6	2.6/15
mfSL4	(1.43, 1.63, 4.6)	1.58	1.4	1.39	507	2,001.2	4.2/15
mfSL6	(0.97, 1.25, 7.25)	0.99	0.95	1.45	624.5	2,660.9	7.9/15
adv	(3.81, 4.34, 11.88)	3.24	3.39	4.68	530	7,608.6	6/15

In order to assess the significance of the obtained experimental results we provide the statistical analysis. There are many statistical techniques for comparing algorithms in literature. We mention Hung and Hong (2009) where SVMr optimized by ant colony optimization algorithm is compared to other methods for the exchange rate forecasting problem. Here, the authors apply the statistical procedure introduced by Diebold and Mariano (2002), which is specialized for determination of prediction accuracy differences in forecasting problems. Unfortunately, the forecast statistical measures do not comply with our setting, i.e., classification problem. Therefore, we followed the statistical analysis methodology similar to one used in Filipović et al. (2013). The Shapiro-Wilk test is carried out in order to investigate the normality of scaled errors. In all three small experimental collections of data, Shapiro-Wilk test showed that data does not follow normal distribution. This excluded the ANOVA test from further consideration, so the non-parametric Kruskal-Wallis H test was applied. In regards to the first experiment, the null hypothesis stated that there is no significant difference between KL, VNS and EM performances. The statistical test showed that null hypothesis should be rejected with $H(2) = 11.68$, $p = 0.003$. It also showed the following mean ranks of the compared algorithms: rank of 25.46 for KL, 22.69 for VNS and 11.85 for EM. For further analysis, KL algorithm was excluded, and Kruskal-Wallis test was applied to VNS and EM. As before, the null hypothesis stated that there is no significant difference between the compared algorithms. Once again, the null hypothesis was rejected with $H(1) = 6.357$, $p = 0.012$, and mean ranks of 17 for VNS and 10 for EM. Similarly, in the second experiment, a significant difference was determined when comparing GS, ACO and EM, with $H(2) = 12.149$, $p = 0.002$ and mean ranks of 12, 9 and 3 for GS, ACO and EM respectively. Further comparison of ACO and EM showed that there was significant difference between these two algorithms, with $H(1) = 7.813$, $p = 0.005$ and mean ranks of 3 for EM AND 8 for ACO. Finally, in the third experimental collection, the null hypothesis considered three algorithms, namely: GS, ES and EM to be statistically equal. As in the first two experiments, the hypothesis was rejected with $H(2) = 22.200$, $p = 0.00002$, and mean ranks of: 34.50 for GS, 21.20 for ES and 13.30 for EM. The second phase statistical testing showed that EM is significantly better than ES with $H(1) = 6.957$, $p = 0.008$, and mean ranks of 19.33 for ES, and 11.67 for EM. Based on the performed statistical tests we conclude that EM is significantly better than the compared algorithms in all three experiments on small instances.

For the large scale experiments, i.e. MKL setting, the reported experimental results of the comparison algorithms are aggregated values, i.e. they represent best, worst and medium classification errors throughout the collection of several algorithms. Therefore, statistical analysis was not performed for those instances.

5 Conclusion

The prediction accuracy of 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. This paper introduces an efficient SVM parameter tuning algorithm based on the electromagnetism-like (EM) optimization.

The real-valued representation of EM points seems to be very well suited in this case, since the decoding procedure consists of scaling from one real domain to another. This allows *smooth* transition through search space and consequently produces high quality solutions. The EM algorithm uses heuristic-based initialization procedure which reduces the time needed to move search towards promising solution regions. Objective function, calculated by using cross validation technique on the training set, shows to be a good estimation of the classification error on testing data. Proficient LS algorithm is applied only on the selected high quality solution points. This cautious use of LS prevents the algorithm from getting trapped in local optima.

In the experimental phase, several diverse data sets and evaluation models were used. Three independent collections of small and medium size test instances with up to 60 features were used to compare EM to the state-of-the-art algorithms from literature. SKL based on radial basis function is used in these experiments. The proposed EM algorithm outperformed the heuristic grid search and variable neighborhood algorithms in 10 out of 13 benchmarks, while it showed to be

superior in comparison to the grid search and evolutionary strategy algorithm by outperforming them in 13 out of 15 cases. In comparison to ant colony optimization technique, EM was also better in 5 out of 5 cases.

Large scale testing is conducted on heterogenous data sets with up to 1,554 features. The results indicated that the proposed method is better than 14 successfull methods in 3 out of 5 cases where RBF multiple kernel learining is used, and competitive in case when linear kernels are used. This led to a conclusion that the EM algorithm is more suitable when dealing with RBF kernels.

As a direction for future work, different local search strategies could be applied, e.g. LS based on grid search with dynamic precision adjustment, random-based LS, etc.

Acknowledgments This work is supported by the Ministry of Education, Science and Technological Development, Republic of Serbia under Grant Numbers: 174010, 174021 and 44006. The authors would like to thank Tanasanee Phienthrakul and Boonserm Kijsirikul for making their benchmark data sets available.

References

Ali M, Golalikhani M (2010) An electromagnetism-like method for nonlinearly constrained global optimization. Comput Math Appl 60(8):2279–2285

Allwein EL, Schapire RE, Singer Y (2001) Reducing multiclass to binary: a unifying approach for margin classifiers. J Mach Learn Res 1:113–141

Aydin I, Karakose M, Akin E (2011) A multi-objective artificial immune algorithm for parameter optimization in support vector machine. Appl Soft Comput 11(1):120–129

Bach FR, Lanckriet GRG, Jordan MI (2004) Multiple kernel learning, conic duality, and the SMO algorithm. In: Proceeding of 21st International Conference on Machine Learning, ACM, New York, NY, USA, ICML '04, pp 6–6

Barbero Jiménez A, López Lázaro J, Dorronsoro JR (2009) Finding optimal model parameters by deterministic and annealed focused grid search. Neurocomput 72(13–15):2824–2832

Birbil SI, Fang SC (2003) An electromagnetism-like mechanism for global optimization. J Global Optim 25:263–282

Birbil SI, Fang SC, Sheu RL (2004) On the convergence of a population-based global optimization algorithm. J Global Optim 30:301–318

Boser BE, Guyon IM, Vapnik VN (1992) A training algorithm for optimal margin classifiers. In: Proceedings of 5th Annual ACM Workshop Computational Learning Theory, ACM Press, pp 144–152

Campbell C, Ying Y (2011) Learning with support vector machines. Synth Lect Artif Intell Mach Learn 5(1):1–95

Carrizosa E, Martín-Barragán B, Romero Morales D (2012) Variable neighborhood search for parameter tuning in support vector machines. Tech. rep.

Chapelle O, Vapnik V, Bousquet O, Mukherjee S (2002) Choosing multiple parameters for support vector machines. Mach Learn 46:131–159

Conforti D, Guido R (2010) Kernel based support vector machine via semidefinite programming: application to medical diagnosis. Comput Oper Res 37(8):1389–1394

Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20:273–297

Črepinšek M, Liu SH, Mernik L (2012) A note on teaching-learning-based optimization algorithm. Inform Sci 212:79–93

Cuevas E, Oliva D, Zaldivar D, Prez-Cisneros M, Sossa H, (2012) Circle detection using electro-magnetism optimization. Inf Sci 182(1):40–55

Diebold FX, Mariano RS (2002) Comparing predictive accuracy. J Bus Econ Stat 20(1)

Duan K, Keerthi S, Poo AN (2003) Evaluation of simple performance measures for tuning svm hyperparameters. Neurocomput 51:41–59

Filipović V (2011) An electromagnetism metaheuristic for the uncapacitated multiple allocation hub location problem. Serdica J Comput 5(3):261–272

Filipović V, Kartelj A, Matić D (2013) An electromagnetism metaheuristic for solving the maximum betweenness problem. Appl Soft Comput 13(2):1303–1313

Franc V, Hlaváč V (2003) Greedy algorithm for a training set reduction in the kernel methods. In: Computer Analysis of Image and Pattern. Springer, Berlin, pp 426–433

Frank A, Asuncion A (2010) UCI machine learning repository. http://archive.ics.uci.edu/ml

Friedrichs F, Igel C (2005) Evolutionary tuning of multiple SVM parameters. Neurocomput 64:107–117

Gascón-Moreno J, Ortiz-García E, Salcedo-Sanz S, Paniagua-Tineo A, Saavedra-Moreno B, Portilla-Figueras J (2011) Multi-parametric gaussian kernel function optimization for ε-SVMr using a genetic algorithm. Adv Comput Intell 113–120

Gascón-Moreno J, Ortiz-García E, Salcedo-Sanz S, Carro-Calvo L, Saavedra-Moreno B, Portilla-Figueras A (2013) Evolutionary optimization of multi-parametric kernel ε-SVMr for forecasting problems. Soft Comput 17(2):213–221

Gascón-Moreno J, Ortiz-García E, Salcedo-Sanz S, Paniagua-Tineo A, Saavedra-Moreno B, Portilla-Figueras J (2013) Multi-parametric gaussian kernel function optimization for ε-SVM$_r$ using a genetic algorithm. In: Cabestany J, Rojas I, Joya G (eds) Advances in Computational Intelligence. Lecture notes in computer science, 6692, Springer, Heidelberg, pp 113–120

Hung WM, Hong WC (2009) Application of svr with improved ant colony optimization algorithms in exchange rate forecasting. Control Cybern 38(3):863–891

Imbault F, Lebart K (2004) A stochastic optimization approach for parameter tuning of support vector machines. In: Proceedings of 17th International Conference on Pattern Recognition, vol 4, pp 597–600

Joachims T (2000) Estimating the generalization performance of an SVM efficiently. In: Proc of the 17th International Conference on Machine Learning

Kartelj A (2012) Electromagnetism metaheuristic algorithm for solving the strong minimum energy topology problem. Yug J Oper Res 22(2)

Kecman V (2001) Learning and soft computing: support vector machines, neural networks, and fuzzy logic models. MIT press, Cambridge

Keerthi S (2002) Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. IEEE Trans Neural Netw 13(5):1225–1229

Keerthi SS, Lin CJ (2003) Asymptotic behaviors of support vector machines with gaussian kernel. Neural Comput 15(7):1667–1689

Lavesson N, Davidsson P (2006) Quantifying the impact of learning algorithm parameter tuning. In: Proceedings of the 21st National Conference on Artificial Intelligence, pp 395–400

Luntz A, Brailovsky V (1969) On estimation of characters obtained in statistical procedure of recognition. Technicheskaya Kibernetica 3

Muller KR, Mika S, Ratsch G, Tsuda K, Scholkopf B (2001) An introduction to kernel-based learning algorithms. IEEE Trans Neural Netw 12(2):181–201

Naji-Azimi Z, Toth P, Galli L (2010) An electromagnetism metaheuristic for the unicost set covering problem. Eur J Oper Res 205(2):290–300

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

Rätsch G, Onoda T, Müller KR (2001) Soft margins for adaboost. Mach Learn 42:287–320

Samadzadegan F, Soleymani A, Abbaspour R (2010) Evaluation of genetic algorithms for tuning svm parameters in multi-class problems. In: Proceedings of the 11th International Symposium on Computational Intellingence and Information, pp 323–328

Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge university press, Cambridge

Sonnenburg S, Rätsch G, Schäfer C, Schölkopf B (2006) Large scale multiple kernel learning. J Mach Learn Res 7:1531–1565

Su CT, Lin HC (2011) Applying electromagnetism-like mechanism for feature selection. Inf Sci 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 sequence-dependent setup times to minimize total weighted tardiness. Soft Comput 13(10):995–1006

Vapnik V (1995) The nature of statistical learning theory. Springer, Berlin

Vapnik VN (1999) An overview of statistical learning theory. IEEE Trans Neural Netw 10(5):988–999

Yurtkuran A, Emel E (2010) A new hybrid electromagnetism-like algorithm for capacitated vehicle routing problems. Expert Syst Appl 37(4):3427–3433

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

Zhiqiang G, Huaiqing W, Quan L (2013) Financial time series forecasting using LPP and SVM optimized by PSO. Soft Comput 17(5):805–818