Journal of the Operational Research Society (2008), 1–7 © 2008 Operational Research Society Ltd. All rights reserved. 0160-5682/08

www.palgrave-journals.com/jors

Support vector machine learning with an evolutionary engine

R Stoean[1]*, M Preuss[2], C Stoean[1], E El-Darzi[3] and D Dumitrescu[4]

[1]University of Craiova, Craiova, Romania; [2]University of Dortmund, Dortmund, Germany; [3]University of Westminster, London, UK; and [4]University of Cluj-Napoca, Cluj, Romania

The paper presents a novel evolutionary technique constructed as an alternative of the standard support vector machines architecture. The approach adopts the learning strategy of the latter but aims to simplify and generalize its training, by offering a transparent substitute to the initial black-box. Contrary to the canonical technique, the evolutionary approach can at all times explicitly acquire the coefficients of the decision function, without any further constraints. Moreover, in order to converge, the evolutionary method does not require the positive (semi-)definition properties for kernels within nonlinear learning. Several potential structures, enhancements and additions are proposed, tested and confirmed using available benchmarking test problems. Computational results show the validity of the new approach in terms of runtime, prediction accuracy and flexibility.

Journal of the Operational Research Society advance online publication, 19 November 2008
doi:10.1057/jors.2008.124

Keywords: evolutionary algorithms; support vector machines; classification; regression

Introduction

Support vector machines (SVMs) have proven to be a powerful tool for classification and regression (Vapnik, 1998; Hsu and Lin, 2002; Mierswa, 2006). Despite the originality and performance of the learning vision, the inner training engine appears as intricate, constrained, rarely transparent and able to converge only for certain particular decision functions. This has motivated us to investigate an alternative training approach, based on evolutionary algorithms (EAs) (Eiben and Smith, 2003), which are known to be flexible and robust.

There have been numerous attempts to combine SVMs and EAs; however, the proposed technique differs from the reported ones. Within the evolutionary approach to support vector machines (EASVMs), the learning path remains unchanged, but the coefficients of the decision function can now be evolved with respect to the optimization objectives regarding accuracy and generalization. The aim of this paper is to overcome the above-stated SVM drawbacks. Apart from the theoretical reasons, EASVMs offer a straightforward and efficient tool for solving practical problems (Stoean *et al*, 2006, 2007).

The paper is structured as follows. The next section presents the ideas and mechanisms of canonical SVMs for classification and regression. This is followed by an outline of our approach including the various interpretations and settings of the parameters. Two alternative EA constructions for the EASVMs are presented and results are compared to those of SVMs. Conclusions are drawn in the final section.

Support vector machines

SVMs are well suited for classification and regression. Given $\{(x_i, y_i)\}_{i=1,2,\ldots,m}$ a training set, where every $x_i \in R^n$ is a data sample and each y_i is a target, a learning task, in general, is concerned with the discovery of the optimal function that minimizes the discrepancy between the given targets of data samples and the predicted ones.

The task for classification is to achieve an optimal separation of given data into classes. SVMs regard learning in this situation from a geometrical point of view: They assume the existence of a separating surface between every two classes labelled as -1 and 1. If training data were linearly separable, then there would exist a linear hyperplane, $\langle w, x\rangle - b = 0$, which partitions the samples according to classes. Separation is achieved if each positive/negative sample lies on the corresponding side of a matching supporting hyperplane for the respective class (1) (Bosch and Smith, 1998).

$$y_i(\langle w, x_i\rangle - b) > 1, \quad i = 1, 2, \ldots, m \tag{1}$$

Hence, SVMs must determine the optimal values for the coefficients of the decision hyperplane such that a good training separation and a high generalization ability, that is, maximal margin between classes (minimal $\|w\|^2$) (Vapnik, 1998), are achieved.

The relative position of samples is not usually suitable for a linear separation; however, training error may be minimized through the introduction of slack variables within the

*Correspondence: *R Stoean, University of Craiova, A.I. Cuza, No 13, 200585, Craiova, Romania.*
E-mail: ruxandra.stoean@inf.ucv.ro

separability statement (1). This relaxation can be achieved by observing the deviation of every data sample from the corresponding supporting hyperplane, which corresponds to a value of $\pm\xi_i/\|w\|$, $\xi_i \geqslant 0$; a ξ_i higher than unity signals an error (Cortes and Vapnik, 1995). The standard algorithm of acquiring the optimal hyperplane relies on an extension of the Lagrange multipliers technique. A dual formulation of the learning task is derived and the optimal Lagrange multipliers are considered as the solutions of the system by setting the gradient of the new objective function to 0.

If a linear separation does not provide acceptable results for classification, then a nonlinear decision surface can be employed by mapping the initial input space into a higher dimensional feature space, $\Phi : R^n \rightarrow H$, where a linear hyperplane can be subsequently derived with the same requirements as before. As in the standard SVM training algorithm, vectors appear only as part of scalar products; the issue can be further simplified by substituting the scalar product by a kernel, which is a function with the property that $K(x_i, x_j) = \langle(\Phi x_i), \Phi(x_j)\rangle$, $x_i, x_j \in R^n$; this can be perceived as to express the similarity between samples. SVMs require the kernel to be a positive (semi-)definite function in order for the standard approach to find a solution to the optimization problem (Mercer's theorem). The problem with this restriction is twofold: First, it is very difficult to check for a newly constructed function and, second, kernels that fail the theorem could otherwise prove to achieve a better separation of the training samples (Mierswa, 2006). Hence, SVMs consequently use a couple of classical kernels that had been demonstrated to meet this prerequisite: The polynomial $K(x, y) = \langle x, y\rangle^p$ and the radial $K(x, y) = e^{\|x-y\|^2/\sigma}$. However, as a substitute for the original solving, a direct search algorithm does not depend on the condition whether the kernel is positive (semi-)definite or not. Having solved the optimization problem, the side of the decision boundary (the class) on which every new data sample lies can now be determined from the derived hyperplane. As it is not always possible to know the map Φ and, as a consequence of the standard methodology, neither to explicitly obtain the coefficients, the class may result from further artifices.

On the other hand, SVMs for regression must find a function, $f(x) = \langle w, x\rangle - b$, that leads to at most ε deviation from the actual targets of data samples (2).

$$y_i - \langle w, x_i\rangle + b \leqslant \varepsilon \quad \text{and} \quad \langle w, x_i\rangle - b - y_i \leqslant \varepsilon, \quad i = 1, 2, \ldots, m \tag{2}$$

Also, $f(x)$ must be as flat as possible (Smola and Scholkopf, 1998) (minimal $\|w\|^2$), since the resulting values of the regression coefficients may affect the model in the sense that it fits current training data but has low generalization ability. The procedure to determine the optimal coefficients of the regression hyperplane follows the same steps as in the SVMs for classification, with an added extra variable ξ_i^* in the condition for approximation of training data. Finally, the predicted target for a test sample is computed following the derived function; the regression coefficients are again rarely transparent and the predicted target is commonly derived from supplementary computations.

An evolutionary alternative training

Evolutionary optimization allows the adaptation of the decision hyperplane to the available training data, which therefore can be treated directly as the (primal) optimization problem. Basic geometric idea of SVMs is considered, but the proposed approach deviates from the standard mathematical treatment. Additionally, the evolutionary technique opens the way for generalizations involving nonlinear, nonstandard decision surfaces. We thus propose a new approach where learning follows the standard SVMs, while the optimal values for the coefficients of the hyperplane (w and b) are directly determined by an EA with respect to the equilibrium between accuracy and generalization ability (Stoean *et al*, 2006, 2007). Although the suggested representation appears to be straightforward, determining other algorithm details, such as interpretation, operators and parameters, is not.

There are other reported attempts to hybridize SVMs and EAs. For example, model selection concerning the adjustment of SVM hyperparameters (free parameters), that is, the penalty for errors and parameters of the kernel, is standardly performed through grid search or gradient descent methods. However, evolution of hyperparameters can be instead achieved through evolution strategies (Friedrichs and Igel, 2004). Evolution of kernel functions to model training data can be performed by means of genetic programming (Howley and Madden, 2005). Finally, the Lagrange multipliers of the dual problem can be evolved by means of evolution strategies and particle swarm optimization (Mierswa, 2006). Inspired by the geometrical SVM learning, Jun and Oh (2006) report the evolution of w and C while using erroneous learning ratio and lift values as the objective function. This paper, however, focuses on the evolution of the coefficients of the decision function within the geometrical learning concept of SVMs.

The EASVM algorithm

The general primal problem of finding the decision hyperplane is consequently solved using an EA. The evolutionary elements (Eiben and Smith, 2003) are set out below.

Representation: The coefficients of the hyperplane are encoded in the structure of an individual: $c = (w_1, \ldots, w_n, b)$. Individuals are initially randomly generated such that $w_i \in [-1, 1]$, $i = 1, 2, \ldots, n$, $b \in [-1, 1]$.

Fitness assignment: The fitness assignment derives from the objective function and is subject to the constraints of the optimization problem. By departing from the standard SVMs, a different nonlinear formulation is derived. The parameter w is mapped through Φ into H. As a result, the squared norm that is involved in the generalization condition becomes $\|\Phi(w)\|^2$ and the equation of the hyper-plane is $\langle\Phi(w), \Phi(x_i)\rangle - b = 0$. The form $\langle u, w\rangle = u^{\mathrm{T}}w$ is used and the kernel is employed to transform the norm in its simplistic equivalence to a scalar product. The fitness formulation (to be minimised) embodies the objective function and the constraints are handled by penalizing the infeasible individuals through a function t that returns the value of the argument, if negative, and 0 otherwise. Its expression for classification is (3):

$$f(w,b) = K(w,w) + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}[t(y_i(K(w,x_i)-b)-1+\xi_i)]^2 \quad (3)$$

while is defined for regression as (4):

$$f(w,b) = K(w,w) + C\sum_{i=1}^{m}(\xi_i+\xi_i^*) + \sum_{i=1}^{m}[t(\varepsilon+\xi_i-y_i+K(w,x_i)-b)]^2 + \sum_{i=1}^{m}[t(\varepsilon+\xi_i^*+y_i-K(w,x_i)+b)]^2 \quad (4)$$

Selection and variation operators: Widely used schemes for real encoding were applied. These are: tournament selection, intermediate crossover and mutation with normal perturbation.

Stop condition: The algorithm stops after a predefined number of generations. Once the near optimal values for the coefficients of the decision hyperplane are found, the target for a new, unseen test data sample can be determined directly by the following equations: (5) for classification or (6) for regression.

$$class(x_i) = sgn(K(w,x_i)-b) \quad (5)$$

$$y_i = K(w,x_i)-b \quad (6)$$

The classification accuracy is defined as the number of correctly labelled cases over the total number of test samples, while regression performance is verified by computing the root mean squared error on the p examples in the test set (7).

$$RMSE = \sqrt{\frac{1}{P}\sum_{i=1}^{p}(y_i^{(\text{pred})}-y_i)^2} \quad (7)$$

A naïve construction

We want to evaluate whether the EASVM algorithm can produce good results when compared to the standard SVM approach. However, first we need to determine appropriate parameters for the algorithm. Also, since indicators for errors, $\xi_i, i = 1, 2, \ldots, m$, appear in the conditions for hyperplane optimality, the issue of their treatment still remains to be addressed. One may depart from the SVM geometrical strict meaning of a deviation and simply evolve the factors of indicators for errors. Hence, EASVMs can handle them through inclusion in the structure of an individual, that is, $c = (w_l, \ldots, w_n, b, \xi_1, \ldots, \xi_m)$, where $\xi_j \in [0, 1]$, $j = 1, 2, \ldots, m$.

Five real-world test problems from the UCI Repository of Machine Learning Databases are used in our experiments. These are diabetes mellitus diagnosis, spam detection, iris recognition, soybean disease diagnosis and Boston housing. The motivation for our choices of test cases was manifold. Diabetes and spam are two-class problems, while soybean and iris are multi-class. Differentiating diabetes diagnosis is a well-known problem; however, spam filtering is a current issue of major concern. Moreover, the latter has a lot more features and samples, which makes a huge difference for classification as well as optimization. Conversely, while soybean has a high number of attributes, iris has only four, but a larger number of samples. Finally, Boston housing is well suited for the regression task. The selected tasks contain all the necessary conditions for the purpose of validating the EASVM approach. Rows 2–5 of Table 1 summarize the test problems.

The experimental design is set as follows. Holdout cross-validation is performed for each data set: 30 runs of the EASVM were generated and, in every run, approximately 70% random cases were assigned to the training set and the remaining 30% went into the test set. Experiments showed the necessity for data normalization in diabetes, spam and iris. SVM hyperparameters are manually chosen and can be found in row 6 of Table 1. The error penalty C was invariably set to 1. For certain (eg radial, polynomial) kernels, the optimization problem shall be relatively simple, due to Mercer's theorem, and is implicitly solved by SVMs. Note that EASVMs are not restricted to using these traditional kernels, but we solely employ them to enable us to compare our algorithm with the classical SVMs. For the iris data set, a radial kernel was used; for diabetes, a polynomial one was employed, while for spam, soybean and Boston, we applied a linear surface. For the multi-class case of iris, the transformed commonly used 1–a–1 method (Hsu and Lin, 2002) was utilized. In the regression case, ε was set to 0. Manually determined EA parameter values are given within the naïve section in rows 8–15 of Table 1. In order to validate the found values for the EA parameters, the semi-automated tuning method of sequential parameter optimization (SPO) (Beielstein, 2006) was applied. The SPO builds on a quadratic regression model, supported by a Latin hypercube sampling (LHS) methodology and noise reduction, by incrementally increased repetition of runs. The best parameter configurations derived by the SPO are depicted in Table 1, under the naïve section, rows 23–30. Test accuracies obtained by both manual (rows 2–7) and SPO tunings (rows 15–20) are presented in Table 2 and the best results are indicated in bold. The automated tuning section shows performances and standard deviations generated by 30 validation runs for the best found configurations of an initial LHS and of the SPO.

The SPO indicates for all the test problems, except for the soybean data set, that crossover probabilities were

Table 1 Data sets properties; manually and SPO tuned parameter values

	Diabetes	*Iris*	*Soybean*	*Spam*	*Boston*
Data					
Number of samples	768	150	47	4601	506
Number of attributes	8	4	35	57	13
Number of classes	2	3	4	2	–
p or σ	$p=2$	$\sigma=1$	$p=1$	$p=1$	$p=1$
Manual tuning					
Naïve representation					
Population size	100	100	100	100	200
Generations	250	100	100	250	2000
Crossover prob.	0.40	0.30	0.30	0.30	0.50
Mutation prob.	0.40	0.50	0.50	0.50	0.50
ξ mutation prob.	0.50	0.50	0.50	0.50	0.50
Mutation strength	0.10	0.10	0.10	0.10	0.10
ξ mutation strength	0.10	0.10	0.10	0.10	0.10
Pruned representation					
Population size	100	100	100	150	200
Generations	250	100	100	300	2000
Crossover prob.	0.4	0.30	0.30	0.80	0.50
Mutation prob.	0.4	0.50	0.50	0.50	0.50
Mutation strength	0.1	4	0.1	3.5	0.1
SPO tuning					
Naïve representation					
Population size	198	46	162	154	89
Generations	296	220	293	287	1755
Crossover prob.	0.87	0.77	0.04	0.84	0.36
Mutation prob.	0.21	0.57	0.39	0.20	0.5
ξ mutation prob.	0.20	0.02	0.09	0.07	0.47
Mutation strength	4.11	4.04	0.16	3.32	0.51
ξ mutation strength	0.02	3.11	3.80	0.01	0.12
Pruned representation					
Population size	190	17	86	11	100
Generations	238	190	118	254	1454
Crossover prob.	0.13	0.99	0.26	0.06	0.88
Mutation prob.	0.58	0.89	0.97	0.03	0.39
Mutation strength	0.15	3.97	0.08	2.58	1.36

dramatically increased, while mutation probabilities were often reduced, especially for errors. However, the relative quality of SPO's final best configurations against the ones found during the initial LHS phase increases with problem size. It must be stated that, in most cases, results achieved with the manually determined parameter values can be improved by SPO—if at all—only by increasing effort, that is, by raising the population size or the number of generations. This brings another benefit for the direct EASVM training: The parameters of the EA can be easily tuned, which is an important aspect when suggesting an alternative solution by means of such algorithms.

The naïve construction produces equally good results when comparing with the canonical SVMs (Table 2, rows 27–32). However, the smaller standard deviations prove the higher stability of the EASVM approach. It must also be remarked that, for the standard kernels, one cannot expect EASVMs to be considerably better than the standard SVMs, since the kernel transformation that induces learning is the same. However, the flexibility of the EAs as optimization tools make EASVMs an attractive choice from the performance perspective, due to their prospective ability to additionally evolve problem-tailored kernels, regardless of whether they are positive (semi-)definite or not, which is impossible within SVMs. However, for large data sets, such as spam filtering, the amount of runtime needed for training is very large. This stems from the large employed genomes, as indicators for errors of every sample in the training set are included in the representation. This problem is resolved by a chunking procedure and resulted in the algorithm running eight times faster than the previous one, at a cost of a small loss in accuracy. Besides solving the EA genome length problem, the proposed mechanism reduces the large number of computations required for referencing the many training samples in the expression of the fitness function.

Table 2 Accuracy/RMSE of EASVMs and SVMs

Manual tuning	*Average*	*Worst*	*Best*	*StD*
Naïve representation				
Diabetes	**76.30**	71.35	80.73	2.24
Iris	**95.18**	91.11	100.0	2.48
Soybean	**99.02**	94.11	100.0	2.23
Spam	**87.74**	85.74	89.83	1.06
Boston	**4.78**	5.95	3.96	0.59
Pruned representation				
Diabetes	**74.60**	70.31	82.81	2.98
Iris	**95.11**	73.33	100	4.83
Soybean	**99.60**	94.12	100	1.49
Spam (overall)	**85.68**	82	88.26	1.72
Boston	**5.07**	6.28	3.95	0.59
SPO tuning	*LHS_{best}*	*StD*	*SPO*	*StD*
Naïve representation				
Diabetes	75.82	3.27	**77.31**	2.45
Iris	**95.11**	2.95	**95.11**	2.95
Soybean	99.61	1.47	**99.80**	1.06
Spam	89.27	1.37	**91.04**	0.80
Boston	5.41	0.65	**5.04**	0.52
Pruned representation				
Diabetes	72.50	2.64	**73.39**	2.82
Iris	**95.41**	2.36	**95.41**	2.43
Soybean	**99.61**	1.47	99.02	4.32
Spam	89.20	1.16	**89.51**	1.17
Boston	4.99	0.66	**4.83**	0.45
SVM results	*Average*	*Worst*	*Best*	*StD*
Diabetes	**76.82**	73.96	81.77	1.84
Iris	**95.33**	88	100	3.16
Soybean	**92.22**	60	100	9.60
Spam	**92.67**	91.56	93.91	0.64
Boston	**3.82**	5.6	2.53	0.7

A simplified EASVM

Although the EASVM provides a viable alternative to SVM, it can still be improved regarding simplicity. The current optimization problem requires to treat the error values, which in the present EA variant are included in the representation. These can severely complicate the problem by increasing the genome length (variable count) by the number of training samples. Moreover, such a methodology strongly drifts away from the canonical SVM concept. In this section, we outline a procedure to represent only the hyperplane coefficients and compute the indicators for errors instead of evolving them, while still maintaining performance.

A pruned construction

Since EASVMs directly and interactively provide hyperplane coefficients at all times, we propose to drop the indicators for errors from the EA representation and, instead, calculate their values. Consequently, individual representation contains only w and b. Additionally, all indicators ξ_i, $i = 1, 2, \ldots, m$, will have to be computed in order to be referred to in the fitness function. For classification, the current individual (separating hyperplane) is taken and supporting hyperplanes are determined through the mechanism in Bosch and Smith (1998). We first compute $m_1 = \min\{K(w, x_i)|y_i = +1\}$ and $m_2 = \max\{K(w, x_i)|y_i = -1\}, i = 1, 2, \ldots, m$. Subsequently, we get $p = |m_1 - m_2|$, while $w' = (2/p)w$ and $b' = (1/p)(m_1 + m_2)$. For every training sample x_i, $i = 1, 2, \ldots, m$, the deviation to its corresponding supporting hyperplane is obtained as either $\delta(x_i) = K(w', x_i) - b' - 1$, $y_i = +1$ or $\delta(x_i) = K(w', x_i) - b' + 1$, $y_i = -1$. If the sign of deviation is equal to class, then the corresponding $\xi_i = 0$; otherwise, the (normalized) absolute deviation is returned as the indicator for error. Experiments showed the need for normalization of the computed deviations in the cases of diabetes, spam and iris, as for a large number of samples, the sum of indicators takes over the whole fitness value. The form of the fitness function remains as in (3), obviously without taking the ξ_is as arguments.

In the case of regression, for every training sample, we calculate the difference between the actual target and the predicted value following the coefficients of the current individual (regression hyperplane), $\delta_i = |K(w, x_i) - b - y_i|$, $i = 1, 2, \ldots, m$. The position against the ε threshold is tested following the condition: If $\delta_i < \varepsilon$, then $\xi_i = 0$, else $\xi_i =$

$\delta_i - \varepsilon$, $i = 1, 2, \ldots, m$. The indicators for errors can now be employed in the fitness evaluation of the corresponding individual, which, due to previous computations, changes from (4) to $f(w, b) = K(w, w) + C\sum_{i=1}^{m} \xi_i$.

Experiments had suggested that the specific method for computing the deviations does not require an additional normalization. The problem settings and SVM parameters are kept the same as in the naïve approach, except for ε which is now set to 5. The manual (rows 16–21) and resulting SPO parameter values (rows 31–36) of the pruned variant are shown in the appropriate section of Table 1, while computational results are depicted in the pruned section of Table 2 (rows 8–13 deriving from manual and rows 21–26 from SPO). Parameter tuning beyond a large initial design appears to be infeasible, as performance is not significantly improved in most cases. This indicates the easiness of parameter setting for the EASVM, because there is a large set of good performing configurations. Nevertheless, there seems to be a slight tendency towards fewer good configurations (harder tuning) for the large problems.

It is interesting to observe that the pruned representation is not that much faster. Although the genome length is drastically reduced, the gained runtime is however partly lost when computing the values for the slack variables, when referring the whole training set once again. The pruned representation, however, has its advantages. Besides featuring smaller genomes, it requires fewer parameters. Since the errors are not evolved anymore, two variables are eliminated. Consequently, this representation is easier to tune. The results of the best configurations for the pruned representation are slightly worse as compared to those recorded for the naïve representation (Table 2). It is noticed that the independent evolution of the slack variables resulted in a better adjustment of the hyperplane as opposed to their strict computation.

For practical consideration, we included a procedure for a dynamic choice of model hyperparameters within the pruned construction. Experiments showed that the parameter expressing the penalty for errors seems of no significance within the EASVM technique. Hence, it is dropped from the parameters pool. Furthermore, by simply inserting one more variable to the genome, the kernel parameter (p or σ) can be evolved. In this way, besides benefiting from the evolutionary approach for solving the primal problem, we are, at the same time, performing model selection. Computational results of the all-inclusive algorithm are similar in accuracy/RMSE to the prior ones, with the exception of an increase in accuracy to 96.45%, in the case of the iris data set. These results are obtained at no additional cost and point to the next extension, the coevolution of non-standard kernels.

Evolutionary engine versus canonical support vector training

In order for a direct comparison to be made, we used the R environment (e1071, mlbench and kernlab packages) for applying SVMs to all data sets. The results, obtained after 30 runs, are illustrated in rows 27–32 of Table 2. After performing manual tuning for the SVM hyperparameters, the best results were obtained for $C = 1$ and $p = 1$, in the cases of diabetes, spam and soybean, and the e1071 default value for $\sigma = 1/m$, for iris and Boston. It is worth noting the difference between the EASVM and the SVM implementations regarding the best performing kernels for the diabetes and Boston problems. The results for each problem were compared via a Wilcoxon rank-sum test. The P-values are as follows: 0.36 for diabetes, 0.84 for iris, 3.98×10^{-5} for soybean, 0.09 for spam and 1.86×10^{-6} for Boston. There are significant differences in the cases of soybean and Boston data sets. However, the absolute difference is not that large for the Boston problem, which indicates that SVM performs slightly better. However, the EASVM outperforms the SVM for the soybean data set.

Although, in terms of accuracy, our approach has not achieved better results for entirely all of the test problems, it has many advantages: The decision surface is always transparent even when working with kernels whose underlying transformation to the feature space cannot be determined. The simplicity of the EA makes the process of solving the classification or regression problem easily explained, better understood and possibly tuned for practical implementation. Most importantly, any function can be used as a kernel and no additional constraints or verifications are necessary. Results show that an evolutionary approach can also be useful in terms of deriving the form of the decision functions. From the opposite perspective, the training is relatively slower than that of SVM, as the evaluation always relates to the training data. However, in practice, it is often the test reaction that is more important. By observing the relationship between each result and the corresponding size of the training data, it is clear that SVM performs better than EASVM for larger problems. This is probably due to the fact that, in these cases, much more evolutionary effort would be necessary in order to achieve a near optimal solution.

Conclusions and outlook

The proposed technique resembles the learning vision of SVMs but has a great advantage in terms of transparency of the training, as it addresses the inherent optimization problem by means of an EA. As opposed to SVMs, EASVMs are much easier to understand and use in practical applications. The unrestricted EASVMs offer a straightforward EA representation and fitness assignment, parameterized by easily tunable values. Moreover, the evolutionary solution of the optimization problem provides directly and within each run all of the function coefficients. The performance of the algorithm is comparable to that of canonical SVMs and in some cases we obtained better results. Two EA representations are used to determine the coefficients: One is simple but fast and the other one is more complicated but more accurate. In order to enhance the efficiency of this approach, a chunking mechanism for reducing the size of large problems is employed and proved to work well. Finally, an all-inclusive EASVM construction, from a practical perspective, is developed and validated.

Although EASVM performs well, further enhancement can still be investigated. As a better choice for treating the two criteria (fit to training data but generalise well), a multicriterial approach can be beneficial. Additionally, the simultaneous evolution of the hyperplane and of nonstandard kernels can be achieved. This approach is highly difficult by means of SVM standard methods for hyperplane determination, whereas it is straightforward for EASVMs.

References

Beielstein T (2006). *Experimental Research in Evolutionary Computation—The New Experimentalism*. Natural Computing. Springer-Verlag: Berlin.

Bosch RA and Smith JA (1998). Separating hyperplanes and the authorship of the disputed federalist papers. *Amer Math Month* **105**(7): 601–608.

Cortes C and Vapnik V (1995). Support vector networks. *J Mach Learn* **20**: 273–297.

Eiben AE and Smith JE (2003). *Introduction to Evolutionary Computing*. Springer-Verlag: Berlin.

Friedrichs F and Igel C (2004). Evolutionary tuning of multiple svm parameters. In: Verleysen M (ed). *Proceedings of the 12th ESANN, Bruges, Belgium*. D-Side Publications: Evere, Belgium, pp 519–524.

Howley T and Madden MG (2005). The genetic kernel support vector machine: Description and evaluation. *Artif Intell Rev* **24**(3–4): 379–395.

Hsu CW and Lin CJ (2002). A comparison of methods for multi-class support vector machines. *IEEE Trans NN* **13**(2): 415–425.

Jun SH and Oh KW (2006). An evolutionary statistical learning theory. *Comput Intell* **3**(3): 249–256.

Mierswa I (2006). *Making indefinite kernel learning practical*. Technical Report, University of Dortmund.

Smola AJ and Scholkopf B (1998). *A tutorial on support vector regression*. Technical Report, University of London.

Stoean R, Preuss M, Dumitrescu D and Stoean C (2006). Evolutionary support vector regression machines. In: O'Conner L (ed). *IEEE Postproceedings of SYNASC, Timisoara, Romania*. IEEE Press: Los Alamitos, CA, pp 330–335.

Stoean R, Preuss M, Stoean C and Dumitrescu D (2007). Concerning the potential of evolutionary support vector machines. In: Srinivasan D and Wang L (eds). *Proceedings of the IEEE CEC, Singapore*. IEEE Press: Piscataway, NJ, pp 1436–1443.

Vapnik V (1998). *Statistical Learning Theory*. Wiley: New York.

Received September 2007;
accepted September 2008 after one revision