Evolutionary Tuning of Multiple SVM Parameters

Frauke Friedrichs, Christian Igel [1]

Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany

Abstract

The problem of model selection for support vector machines (SVMs) is considered. We propose an evolutionary approach to determine multiple SVM hyperparameters: The covariance matrix adaptation evolution strategy (CMA-ES) is used to determine the kernel from a parameterized kernel space and to control the regularization. Our method is applicable to optimize non-differentiable kernel functions and arbitrary model selection criteria. We demonstrate on benchmark datasets that the CMA-ES improves the results achieved by grid search already when applied to few hyperparameters. Further, we show that the CMA-ES is able to handle much more kernel parameters compared to grid-search and that tuning of the scaling and the rotation of Gaussian kernels can lead to better results in comparison to standard Gaussian kernels with a single bandwidth parameter. In particular, more flexibility of the kernel can reduce the number of support vectors.

Key words: support vector machines, model selection, evolutionary algorithms

1 Introduction

Support vector machines (SVMs, e.g., [5,22,24]) are learning machines based on two key elements: a general purpose linear learning algorithm and a problem specific kernel that computes the inner product of input data points in a feature space. The choice of the kernel function is the crucial step in handling a learning task with an SVM. For example, it is important to achieve a distribution of the data in the feature space that reflects the affiliation to the class labels. Often a parameterized family of kernel functions is considered and the problem reduces to finding an appropriate parameter vector for the given problem. In case of noisy, non-separable data one also has to choose a regularization parameter, which controls the trade-off between minimizing the

[1] Christian.Igel@neuroinformatik.rub.de

training error and the complexity of the decision function. The kernel parameters together with the regularization parameter are called the hyperparameters of the SVM.

In practice the hyperparameters are usually determined by grid search. That is, the hyperparameters are varied with a fixed step-size through a wide range of values and the performance of every combination is assessed using some performance measure. Because of the computational complexity, grid search is only suitable for the adjustment of very few parameters. Perhaps the most elaborate systematic technique for choosing multiple hyperparameters are gradient descent methods [3,4,8,15]. These algorithms iterate the following procedure: The SVM is trained using the current hyperparameter vector, the gradient of some generalization error bound w.r.t. the hyperparameters is calculated, and a step is performed in the parameter space based on this gradient. However, this approach has some significant drawbacks. The kernel function has to be differentiable. The score function for assessing the performance of the hyperparameters (or at least an accurate approximation of this function) also has to be differentiable with respect to kernel and regularization parameters. This excludes reasonable measures such as the number of support vectors. In [3, Sec. 6.2] separability of the dataset is assumed when computing the derivative, which is a very restrictive assumption. Iterative gradient-based algorithms, which usually rely on smoothed approximations of a score function, do not ensure that the search direction points exactly to an optimum of the original, often discontinuous generalization performance measure.

We propose an evolutionary method for hyperparameter selection that does not suffer from the limitations described above. Evolutionary algorithms have been successfully applied to model selection for neural networks [11,18,25]. This includes the recent applications of genetic algorithms for feature selection of SVMs [6,7,14,17]. We use the covariance matrix adaptation evolution strategy (CMA-ES, [10]) to search for an appropriate hyperparameter vector. The fitness function that is optimized directly corresponds to some generalization performance measure. We apply our method to tuning Gaussian kernels, where not only the scaling but also the orientation is adapted.

We give a short description of SVMs in Section 2 and of the CMA-ES in Section 3. The parameterization of general Gaussian kernels is introduced in Section 4. We present experimental results in Section 5 and draw our conclusions in Section 6.

2 Support Vector Machines

We consider L_1-norm soft margin SVMs for the discrimination of two classes. Let $(\boldsymbol{x}_i, y_i), 1 \leq i \leq \ell$, be the training examples, where $y_i \in \{-1, 1\}$ is the label associated with input pattern $\boldsymbol{x}_i \in X$. The main idea of SVMs is to map the input vectors to a feature space F and to classify the transformed data by a linear function. The transformation $\phi : X \to F$ is implicitly done by a kernel $K : X \times X \to \mathbb{R}$, which computes an inner product in the feature space, i.e., $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$. The linear function for classification in the feature space is chosen according to a generalization error bound considering a target margin and the margin slack vector, i.e., the amounts by which individual training patterns fail to meet that margin (cf. [5,22,24]). This leads to the SVM decision function $f(\boldsymbol{x}) = \text{sign}\left(\sum_{i=1}^{\ell} y_i \alpha_i^* K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right)$, where the coefficients α_i^* are the solution of the following quadratic optimization problem: Maximize $W(\boldsymbol{\alpha}) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} y_i y_j \alpha_i \alpha_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ subject to $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ and $0 \leq \alpha_i \leq C$ for $i = 1, \ldots, \ell$. The optimal value for b can then be computed based on the solution $\boldsymbol{\alpha}^*$. The vectors $\boldsymbol{x}_i$ with $\alpha_i > 0$ are called support vectors. The regularization parameter C controls the trade-off between maximizing the target margin and minimizing the L_1-norm of the margin slack vector of the training data.

3 Evolution Strategies

Evolution strategies (ES, [2,20,23]) are one of the main branches of evolutionary algorithms, i.e., a class of iterative, direct, randomized optimization methods mimicking principles of neo-Darwinian evolution theory. Here, we use the highly efficient CMA-ES [9,10]. A set of μ individuals that form the parent population is maintained. Each individual has a genotype that encodes a candidate solution for the optimization problem at hand, in this study an m-dimensional real-valued object variable vector representing the SVM hyperparameters. The fitness of an individual is equal to the objective function value at the point in the search space it represents. In each iteration of the algorithm, $\lambda > \mu$ new individuals, the offspring, are generated by partially stochastic variations of parent individuals. The fitness of the offspring is computed and the μ best of the offspring form the next parent population. This loop of variation and selection is repeated until a termination criterion is met.

In the following, we describe the covariance matrix adaptation ES (CMA-ES) proposed in [9,10], which performs efficient real-valued optimization. Each individual represents an m-dimensional real-valued object variable vector. These variables are altered by two variation operators, intermediate recombination and additive Gaussian mutation. The former corresponds to computing the

center of mass of the μ individuals in the parent population. Mutation is realized by adding a normally distributed random vector with zero mean. In the CMA-ES, the complete covariance matrix of the Gaussian mutation distribution is adapted during evolution to improve the search strategy. More formally, the object parameters $\boldsymbol{g}_k^{(t)}$ of offspring $k = 1, \ldots, \lambda$ created in generation t are given by

$$\boldsymbol{g}_k^{(t)} = \langle \tilde{\boldsymbol{g}} \rangle^{(t)} + \sigma^{(t)} \boldsymbol{B}^{(t)} \boldsymbol{D}^{(t)} \boldsymbol{z}_k^{(t)} \quad ,$$

where $\langle \tilde{\boldsymbol{g}} \rangle^{(t)} = \frac{1}{\mu} \sum_{i=1}^{\mu} \tilde{\boldsymbol{g}}_i^{(t)}$ is the center of mass of the parent population in generation t and the $\boldsymbol{z}_k^{(t)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ are independent realizations of an m-dimensional normally distributed random vector with zero mean and covariance matrix equal to the identity matrix $\boldsymbol{I}$. The covariance matrix $\boldsymbol{C}^{(t)}$ of the random vectors

$$\sigma^{(t)} \boldsymbol{B}^{(t)} \boldsymbol{D}^{(t)} \boldsymbol{z}_k^{(t)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(t)})$$

is a symmetric positive $m \times m$ matrix with

$$\boldsymbol{C}'^{(t)} = \boldsymbol{C}^{(t)} / {\sigma^{(t)}}^2 = \boldsymbol{B}^{(t)} \boldsymbol{D}^{(t)} \left(\boldsymbol{B}^{(t)} \boldsymbol{D}^{(t)} \right)^T \quad .$$

The columns of the orthogonal $m \times m$ matrix $\boldsymbol{B}^{(t)}$ are the normalized eigenvectors of $\boldsymbol{C}'^{(t)}$ and $\boldsymbol{D}^{(t)}$ is a $m \times m$ diagonal matrix with the square roots of the corresponding eigenvalues.

The strategy parameters, both the matrix $\boldsymbol{C}'^{(t)}$ and the so called global stepsize $\sigma^{(t)}$, are updated online using the covariance matrix adaptation (CMA) method. The key idea of the CMA is to alter the mutation distribution in a deterministic way such that the probability to reproduce steps in the search space that have led to the current population is increased. This enables the algorithm to detect correlations between object variables and to become invariant under orthogonal transformations of the search space (apart from the initialization). In order to use the information from previous generations efficiently, the search path of the population over a number of past generations is taken into account.

In the CMA-ES, rank-based (μ, λ)-selection is used for environmental selection. That is, the μ best of the λ offspring form the next parent population. After selection, the strategy parameters are updated:

$$\boldsymbol{s}^{(t+1)} = (1 - c) \cdot \boldsymbol{s}^{(t)} + c_{\mathrm{u}} \cdot \underbrace{\sqrt{\mu}\, \boldsymbol{B}^{(t)} \boldsymbol{D}^{(t)} \langle \boldsymbol{z} \rangle_{\mu}^{(t)}}_{\frac{\sqrt{\mu}}{\sigma^{(t)}} \left(\langle \tilde{\boldsymbol{g}} \rangle^{(t+1)} - \langle \tilde{\boldsymbol{g}} \rangle^{(t)} \right)}$$

$$\boldsymbol{C}'^{(t+1)} = (1 - c_{\mathrm{cov}}) \cdot \boldsymbol{C}'^{(t)} + c_{\mathrm{cov}} \cdot \boldsymbol{s}^{(t+1)} \left(\boldsymbol{s}^{(t+1)} \right)^{\mathrm{T}} \quad .$$

Herein, $\boldsymbol{s}^{(t+1)} \in \mathbb{R}^m$ is the evolution path—a weighted sum of the centers of the population over the generations starting from $\boldsymbol{s}^{(1)} = \mathbf{0}$. The factor $\sqrt{\mu}$ compensates for the loss of variance due to computing the center of mass. The parameter $c \in]0, 1]$ controls the time horizon of the adaptation of $\boldsymbol{s}$; we set $c = 1/\sqrt{m}$. The constant $c_{\mathrm{u}} = \sqrt{c(2-c)}$ normalizes the variance of $\boldsymbol{s}$ (viewed as a random variable) as $1^2 = (1-c)^2 + c_{\mathrm{u}}^2$. The expression $\langle \boldsymbol{z} \rangle_{\mu}^{(t)} = \frac{1}{\mu} \sum_{i=1}^{\mu} \boldsymbol{z}_{i:\lambda}^{(t)}$ is the average of the realizations of the random vector that led to the new parent population, where $i{:}\lambda$ denotes the index of the offspring having the ith best fitness value of all offspring in the current generation ($\tilde{\boldsymbol{g}}_i^{(t+1)} = \boldsymbol{g}_{i:\lambda}^{(t)}, i = 1, \ldots, \mu$). The parameter $c_{\mathrm{cov}} \in [0, 1[$ controls the update of $\boldsymbol{C}'^{(t)}$ and we set it to $c_{\mathrm{cov}} = 2/(m^2 + m)$. This update rule shifts $\boldsymbol{C}'^{(t)}$ towards the $m \times m$ matrix $\boldsymbol{s}^{(t+1)} \left(\boldsymbol{s}^{(t+1)}\right)^T$ making mutation steps in the direction of $\boldsymbol{s}^{(t+1)}$ more likely. The evolution path does not only represent the last (adaptive) step of the parent population, but a time average over all previous adaptive steps. The influence of previous steps decays exponentially, where the decay rate is controlled by c.

The adaptation of the global step-size parameter σ is done separately on a shorter timescale (a single parameter can be estimated based on less samples compared to the complete covariance matrix). We keep track of a second evolution path $\boldsymbol{s}_\sigma$ without the scaling by $\boldsymbol{D}$:

$$\boldsymbol{s}_\sigma^{(t+1)} = (1 - c_\sigma) \cdot \boldsymbol{s}_\sigma^{(t)} + c_{\mathrm{u}_\sigma} \cdot \sqrt{\mu}\, \boldsymbol{B}^{(t)} \langle \boldsymbol{z} \rangle_{\mu}^{(t+1)}$$

$$\sigma^{(t+1)} = \sigma^{(t)} \cdot \exp\left(\frac{\|\boldsymbol{s}_\sigma^{(t+1)}\| - \hat{\boldsymbol{\chi}}_n}{d \cdot \hat{\boldsymbol{\chi}}_n} \right) \ ,$$

where $\hat{\boldsymbol{\chi}}_n$ is the expected length of a m-dimensional, normally distributed random vector with covariance matrix $\boldsymbol{I}$. The damping parameter $d \geq 1$ decouples the adaptation rate from the strength of the variation. We set $d = \sqrt{m}$ and start from $\boldsymbol{s}_\sigma^{(1)} = \mathbf{0}$. The parameter $c_\sigma \in]0, 1]$ controls the update of $\boldsymbol{s}_\sigma$. Here, we use $c_\sigma = c$. Setting $c_{\mathrm{u}_\sigma} = \sqrt{c_\sigma(2 - c_\sigma)}$ normalizes the variance of $\boldsymbol{s}_\sigma$. The evolution path $\boldsymbol{s}_\sigma$ is the sum of normally distributed random variables. Because of the normalization, its expected length would be $\hat{\boldsymbol{\chi}}_n$ if there were no selection. Hence, the update rule basically increases the global step-size if the steps leading to selected individuals have been larger than expected and decreases the step size in the opposite case.

The CMA-ES needs only small population sizes. In the following, they are chosen according to the heuristics $\lambda = \max(5, \min(m, 4 + \lfloor 3 \ln m \rfloor))$ and $\mu = \max(1, \lfloor \lambda/4 \rfloor)$ based on [10], where m is the number of object variables. The initializations of $\boldsymbol{C}'^{(1)}$ and $\sigma^{(1)}$ allow for incorporation of prior knowledge about the scaling of the search space (see section 5.1). Note that all other parameters of the algorithm can be set to the carefully determined default values given

in [9,10]. Of course, tuning for example the cumulation by changing c and c_σ for a particular class of cost functions can increase the search performance, but without additional knowledge about the objective function we stick to the robust default values, see [10] for a detailed discussion.

4 Encoding General Gaussian Kernels

We consider SVMs with general Gaussian kernels

$$K_{\boldsymbol{A}}(\boldsymbol{x}, \boldsymbol{z}) = e^{-(\boldsymbol{x}-\boldsymbol{z})^T \boldsymbol{A}(\boldsymbol{x}-\boldsymbol{z})} \ ,$$

where $\boldsymbol{A}$ is a symmetric positive definite matrix. We use three different parameterizations of these kernels yielding three nested spaces of kernel functions. Firstly, there are the ordinary Gaussian kernels $K_{\tau \boldsymbol{I}}$, where $\boldsymbol{I}$ is the unit matrix and $\tau > 0$ the only adjustable parameter. Secondly, we allow independent scalings of the components of the input vectors and consider the kernel family $K_{\boldsymbol{D}}$, where $\boldsymbol{D}$ is a diagonal matrix with positive entries. Finally, we allow arbitrary symmetric positive definite matrices $\boldsymbol{A}$, i.e., the input space can be scaled and rotated. This kernel family is denoted simply by $K_{\boldsymbol{A}}$.

The individuals in the ES encode the regularization parameter C and the kernel parameters. The ES obeys restrictions of parameters to $\mathbb{R}^+$ when necessary (see below). However, when encoding $K_{\boldsymbol{A}}$ kernels, we have to ensure that after variation the genotype still corresponds to a feasible (i.e., symmetric, positive definite) matrix. We use the fact that for any symmetric and positive definite $n \times n$ matrix $\boldsymbol{A}$ there exists an orthogonal $n \times n$ matrix $\boldsymbol{T}$ and a diagonal $n \times n$ matrix $\boldsymbol{D}$ with positive entries such that $\boldsymbol{A} = \boldsymbol{T}^T \boldsymbol{D} \boldsymbol{T}$ and

$$\boldsymbol{T} = \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \boldsymbol{R}(\alpha_{i,j}) \ ,$$

see [21]. The $n \times n$ matrices $\boldsymbol{R}(\alpha_{i,j})$ are elementary rotation matrices which are equal to the unit matrix except for $[\boldsymbol{R}(\alpha_{i,j})]_{ii} = [\boldsymbol{R}(\alpha_{i,j})]_{jj} = \cos \alpha_{ij}$ and $[\boldsymbol{R}(\alpha_{i,j})]_{ji} = -[\boldsymbol{R}(\alpha_{i,j})]_{ij} = \sin \alpha_{ij}$. Each genotype encodes the $n + (n^2 - n)/2 + 1$ parameters

$$(C', d_1, \ldots, d_n, \alpha_{1,2}, \alpha_{1,3}, \ldots, \alpha_{1,n}, \alpha_{2,3}, \alpha_{2,4}, \ldots, \alpha_{2,n}, \ldots, \alpha_{n-1,n}) \ ,$$

and we set $\boldsymbol{D} = \mathrm{diag}(|d_1|, \ldots, |d_n|)$ and $C = |C'|$.

5 Experimental Evaluation

5.1 Experiments

For the evaluation of our hyperparameter optimization method we used the common medical benchmark datasets *Breast-Cancer*, *Diabetes*, *Heart*, and *Thyroid* with input dimension n equal to 9, 8, 13, and 5, and total number of patterns 277, 768, 270, and 215, respectively. The data were preprocessed and partitioned as in [19]. Each component of the input data is normalized to zero mean and unit standard deviation. There are 100 partitions of each dataset into disjoint training and test sets. In [16], appropriate SVM hyperparameters for the Gaussian kernel $K_{\tau \boldsymbol{I}}$ were determined using a two-stage grid-search and the following score function: For each hyperparameter combination, five SVMs are built using the training sets of the first five data partitions and the average of the classification rates on the corresponding five test sets determines the score value (Test-5) of this parameter vector. The hyperparameter vector with the best score is selected and its performance is measured by calculating the score function using all 100 partitions (Test-100). [2]

We used this scenario as a basis for our evolutionary hyperparameter adaptation approach: We took the results of the grid search performed in [16] as initial values for the CMA-ES and used the score function described above to determine the fitness. The initializations of τ and C were approximately 0.00661 and 280, 0.00090 and 274, 0.00120 and 8.8, and 0.05863 and 401 for *Breast-Cancer*, *Diabetes*, *Heart*, and *Thyroid*, respectively. Grid-search reveals that the hyperparameter optimization problem is indeed multi-modal, see Fig. 1 for an example. It is not necessary to determine the starting points for the ES by grid-search. Typically, we start the CMA-ES from a randomly chosen point or in an area which was roughly determined by a very coarse grid-search.

For each benchmark data set and each of the three kernel families, 20 evolutionary optimizations over 250 generations were performed. In each trial, the hyperparameter vector with the best fitness value (Test-5) was regarded as the solution. The performance of each solution was assessed by calculating the score using all 100 data partitions (Test-100). The SVMs were trained using SVM^{light} [13]. We set the parameters of the initial CMA mutation distribution to $\sigma^{(1)} = 1$ and $\boldsymbol{C}'^{(1)} = \mathrm{diag}(0.001, 0.00001, \ldots, , 0.00001)$, because the regularization parameter C should be varied stronger compared to the kernel parameters in the beginning.

[2] We have chosen this evaluation procedure in order to compare our results one-to-one with [16]. As in [16,19], the data for computing Test-5 and Test-100 overlap and hence Test-100 is only a weak indicator for the generalization performance. However, we tested other evaluation criteria which gave qualitatively the same results.

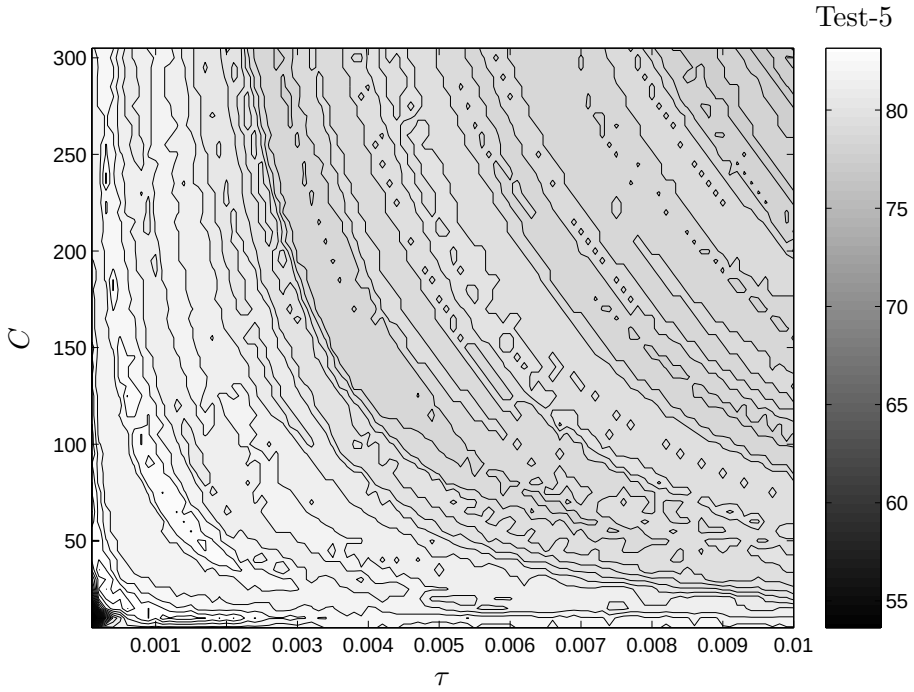

Fig. 1. Score function (average classification rates on the five test sets, Test-5, in percent) for the *Heart* dataset using $K_{\tau \boldsymbol{I}}$ with different combinations of C and τ.

5.2 Results

We compared the start values of Test-5 and Test-100 with the results obtained by the ES, see Table 1. Except for a few cases, we achieved significantly (t-test, $p < 0.05$) better results by evolutionary optimization. The ES could even improve the performance when adapting $K_{\tau \boldsymbol{I}}$ in some cases, although the initial values were already tuned for $K_{\tau \boldsymbol{I}}$ by intensive grid search.

We also used a t-test to compare the performance of the resulting SVMs when the evolutionary search was performed in the three spaces of kernel functions. The scaled kernels $K_{\boldsymbol{D}}$ as well as the scaled and rotated kernels $K_{\boldsymbol{A}}$ led to significantly ($p < 0.05$) better results compared to the ordinary Gaussian kernel $K_{\tau \boldsymbol{I}}$ (except for *Diabetes*: Test-100 with kernel $K_{\boldsymbol{A}}$). The major improvement was already achieved by allowing scaling. However, in one case (*Breast-Cancer*: Test-5) the kernel $K_{\boldsymbol{A}}$ with the additional rotation parameters gave significantly ($p < 0.05$) better results than just scaling.

There is another remarkable advantage of the scaled kernel $K_{\boldsymbol{D}}$ and the scaled and rotated kernel $K_{\boldsymbol{A}}$: The number of support vectors decreases. Evolutionary adaptation of kernels $K_{\boldsymbol{D}}$ and $K_{\boldsymbol{A}}$ led to SVMs having significantly less support vectors compared to the ordinary kernel $K_{\tau \boldsymbol{I}}$ optimized by the ES or grid-search, see Table 1. Again, in one case (*Diabetes*) the kernel $K_{\boldsymbol{A}}$ with additional rotation parameters yielded a significantly ($p < 0.05$) better result than just

Table 1
Results averaged over 20 trials ± standard deviations. The first column specifies the benchmark: *Breast-Cancer* (C), *Diabetes* (D), *Heart* (H), and *Thyroid* (T). The second column indicates whether the results refer to the initial grid search values (init, [16]) or to the evolutionary optimized kernels $K_{\boldsymbol{\tau I}}$, $K_{\boldsymbol{D}}$, or $K_{\boldsymbol{A}}$. The $\hat{t}$ values are the generations needed to evolve the final solution on average and λ is the number of offspring per generation (μ is chosen according to the heuristic given in section 3). Thus, $\hat{t} \cdot \lambda$ indicates the average computational costs to find the final parameters, of course, good values are already found earlier in the evolutionary process. The percentages of correctly classified patterns on the 5 and 100 test sets are given by Test-5 and Test-100. The average numbers of support vectors over the 100 training sets is #SV-100. Results statistically significantly better compared to grid-search (init) are marked with $\star$ (two-sided t-test, $p < 0.05$).

Data	Kernel	$\hat{t}$	λ	Test-5	Test-100	#SV-100
C	init			73.77	74.51	113.52
	$K_{\boldsymbol{\tau I}}$	10 ± 15	5	$74.00 \pm 0.08^\star$	$74.56 \pm 0.04^\star$	114.58 ± 1
	$K_{\boldsymbol{D}}$	83 ± 42	10	$76.91 \pm 0.96^\star$	$75.17 \pm 0.65^\star$	$113.10 \pm 0.89^\star$
	$K_{\boldsymbol{A}}$	179 ± 59	15	$77.47 \pm 0.53^\star$	$75.38 \pm 0.42^\star$	$112.70 \pm 0.68^\star$
D	init			76.34	76.67	247.83
	$K_{\boldsymbol{\tau I}}$	102 ± 78	5	$76.84 \pm 0.11^\star$	76.67 ± 0.04	251.63 ± 8.14
	$K_{\boldsymbol{D}}$	149 ± 56	9	$78.10 \pm 0.22^\star$	$76.88 \pm 0.25^\star$	$239.31 \pm 2.84^\star$
	$K_{\boldsymbol{A}}$	181 ± 56	14	$78.07 \pm 0.19^\star$	76.73 ± 0.32	$235.73 \pm 3.43^\star$
H	init			83.80	84.79	106.33
	$K_{\boldsymbol{\tau I}}$	10 ± 17	5	83.81 ± 0.04	84.74 ± 0.04	$103.37 \pm 0.97^\star$
	$K_{\boldsymbol{D}}$	97 ± 41	11	$85.71 \pm 0.43^\star$	$84.98 \pm 0.36^\star$	$76.52 \pm 2.10^\star$
	$K_{\boldsymbol{A}}$	153 ± 47	17	$85.86 \pm 0.17^\star$	$85.14 \pm 0.33^\star$	$75.51 \pm 1.5^\star$
T	init			96.27	95.83	16.36
	$K_{\boldsymbol{\tau I}}$	3 ± 5	5	$96.56 \pm 0.08^\star$	95.74 ± 0.2	15.99 ± 1.01
	$K_{\boldsymbol{D}}$	16 ± 35	6	$97.29 \pm 0.1^\star$	$96.01 \pm 0.04^\star$	$15.46 \pm 0.12^\star$
	$K_{\boldsymbol{A}}$	5 ± 3	12	$97.33 \pm 0^\star$	$96.01 \pm 0.05^\star$	$15.42 \pm 0.18^\star$

scaling.

6 Conclusions

The CMA evolution strategy is a powerful, general method for SVM hyperparameter selection. It can handle a large number of kernel parameters and does

neither require differentiable kernels and model selection criteria nor separability of the data. As typical selection criteria exhibit multiple local optima (e.g., see Fig. 1 and [8]), we claim that evolutionary optimization is generally better suited for SVM model selection than gradient-based methods.

It is often argued that evolutionary optimization of kernels is too time consuming for real-world applications. Given a certain amount of time for designing the classifier, there is a maximum number of objective function evaluations—and the question is how to spend these evaluations. Grid-search, for example, scales exponentially in the number of objective parameters. In contrast, a simple evolution strategy scales linearly on functions that are monotone with respect to the distance to the optimium [1,12]. The objective functions in case of SVM optimization seem to be well suited for evolution strategies (see Fig. 1), which can follow a global trend that is superposed by local minima. Thus, in problems which are currently addressed by grid-search the number of objective parameters and the accuracy of the optimization can be increased when using an evolutionary algorithm.

We demonstrated that extended Gaussian kernels with scaling (and rotating) parameters can lead to a significantly better performance than standard Gaussian kernels. To our knowledge, for the first time a method that adapts the orientation of Gaussian kernels—i.e., that can detect correlations in the input data space *relevant for the kernel machine*—was presented. Our experiments show that by increasing the flexibility of the kernels the number of support vectors can be reduced. Future work will address other, especially non-differentiable kernels and optimization criteria.

Acknowledgement

We thank the authors of [16] for providing their grid-search results and the reviewers for their careful study of the manuscript.

References

[1] H.-G. Beyer. *The Theory of Evolution Strategies*. Springer-Verlag, 2001.

[2] H.-G. Beyer and H.-P. Schwefel. Evolution strategies: A comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

[3] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1):131–159, 2002.

[4] K.-M. Chung, W.-C. Kao, C.-L. Sun, and C.-J. Lin. Radius margin bounds for support vector machines with RBF kernel. *Neural Computation*, 15(11):2643–2681, 2003.

[5] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[6] D. R. Eads, D. Hill, S. Davis, S. J. Perkins, J. Ma, R. B. Porter, and J. P. Theiler. Genetic algorithms and support vector machines for time series classification. In B. Bosacchi, D. B. Fogel, and J. C. Bezdek, editors, *Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation V.*, volume 4787 of *Proceedings of the SPIE*, pages 74–85, 2002.

[7] H. Fröhlich, O. Chapelle, and B. Schölkopf. Feature selection for support vector machines by means of genetic algorithms. In *15th IEEE International Conference on Tools with AI (ICTAI 2003)*, pages 142–148. IEEE Computer Society, 2003.

[8] C. Gold and P. Sollich. Model selection for support vector machine classification. *Neurocomputing*, 55(1-2):221–249, 2003.

[9] N. Hansen and A. Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu, \lambda)$-CMA-ES. In *5th European Congress on Intelligent Techniques and Soft Computing (EUFIT'97)*, pages 650–654. Aachen, Germany: Verlag Mainz, Wissenschaftsverlag, 1997.

[10] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[11] C. Igel, S. Wiegand, and F. Friedrichs. Evolutionary optimization of neural systems: The use of self-adaptation. In *4th International Meeting on Constructive Approximation (IDoMAT 2004)*. Birkhäuser Verlag. To appear.

[12] J. Jägersküpper. Analysis of a simple evolutionary algorithm for minimization in euclidian spaces. In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP 2003)*, volume 2719 of *LNCS*, pages 1068–1079. Springer Verlag, 2003.

[13] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, chapter 11, pages 169–184. MIT Press, 1999.

[14] K. Jong, E. Marchiori, and A. van der Vaart. Analysis of proteomic pattern data for cancer detection. In G. R. Raidl, S. Cagnoni, J. Branke, D. W. Corne, R. Drechsler, Y. Jin, C. G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G. D. Smith, and G. Squillero, editors, *Applications of Evolutionary Computing*, number 3005 in LNCS, pages 41–51. Springer-Verlag, 2004.

[15] S. S. Keerthi. Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, 2002.

[16] P. Meinicke, T. Twellmann, and H. Ritter. Discriminative densities from maximum contrast estimation. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 985–992, Cambridge, MA, 2002. MIT Press.

[17] M. T. Miller, A. K. Jerebko, J. D. Malley, and R. M. Summers. Feature selection for computer-aided polyp detection using genetic algorithms. In A. V. Clough and A. A. Amini, editors, *Medical Imaging 2003: Physiology and Function: Methods, Systems, and Applications*, volume 5031 of *Proceedings of the SPIE*, pages 102–110, 2003.

[18] S. Nolfi. Evolution and learning in neural networks. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*, pages 415–418. MIT Press, 2 edition, 2002.

[19] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for adaboost. *Machine Learning*, 42(3):287–32, 2001.

[20] I. Rechenberg. *Evolutionsstrategie '94*. Werkstatt Bionik und Evolutionstechnik. Frommann-Holzboog, Stuttgart, 1994.

[21] G. Rudolph. On correlated mutations in evolution strategies. In R. Männer and B. Manderick, editors, *Parallel Problem Solving from Nature 2 (PPSN II)*, pages 105–114. Elsevier, 1992.

[22] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[23] H.-P. Schwefel. *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology Series. John Wiley & Sons, 1995.

[24] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[25] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

Frauke Friedrichs received her diploma in biology in 2001 and her diploma in mathematics in 2003 from the University of Bochum, Germany. In 2003 she joined the Department of Theoretical Biology at the Institute for Neuroinformatics in Bochum. Since 2004 she is a PhD student at the Department of Genetic Epidemiology which belongs to the Institute for Arteriosclerosis Research in Münster, Germany.

Christian Igel received the diploma degree in computer science from the University of Dortmund, Germany, in 1997. Since then he has been with the Institut für Neuroinformatik (INI) at the Ruhr-Universität Bochum, Germany. He received the doctoral degree from the Technical Faculty of the University of Bielefeld, Germany, in 2002. In 2003, he was appointed Juniorprofessor for the Optimization of Adaptive Systems at the INI. His research focuses on biological and technical aspects of neural and evolutionary information processing.