In:
Yaochu Jin (Ed.), *Multi-objective Machine Learning*
Studies in Computational Intelligence, Vol. 16, pp. 199-220, Springer-Verlag, 2006

Multi-objective optimization of support vector machines

Thorsten Suttorp[1] and Christian Igel[2]

[1] thorsten.suttorp@neuroinformatik.rub.de
[2] christian.igel@neuroinformatik.rub.de
Institut für Neuroinformatik
Ruhr-Universität Bochum
44780 Bochum, Germany

Summary. Designing supervised learning systems is in general a multi-objective optimization problem. It requires finding appropriate trade-offs between several objectives, for example between model complexity and accuracy or sensitivity and specificity. We consider the adaptation of kernel and regularization parameters of support vector machines (SVMs) by means of multi-objective evolutionary optimization. Support vector machines are reviewed from the multi-objective perspective, and different encodings and model selection criteria are described. The optimization of split modified radius-margin model selection criteria is demonstrated on benchmark problems. The MOO approach to SVM design is evaluated on a real-world pattern recognition task, namely the real-time detection of pedestrians in infrared images for driver assistance systems. Here the three objectives are the minimization of the false positive rate, the false negative rate, and the number of support vectors to reduce the computational complexity.

1 Introduction

The design of supervised learning systems for classification requires finding a suitable trade-off between several objectives, especially between model complexity and accuracy on a set of noisy training examples ($\rightarrow$ bias vs. variance, capacity vs. empirical risk). In many applications, it is further advisable to consider sensitivity and specificity (i.e., true positive and true negative rate) separately. For example in medical diagnosis, a high false alarm rate may be tolerated if the sensitivity is high. The computational complexity of a solution can be an additional design objective, in particular under real-time constraints.

This multi-objective design problem is usually tackled by aggregating the objectives into a scalar function and applying standard methods to the resulting single-objective task. However, such an approach can only lead to

satisfactory solutions if the aggregation (e.g., a linear weighting of empirical error and regularization term) matches the problem. A better way is to apply "true" multi-objective optimization (MOO) to approximate the set of Pareto-optimal trade-offs and to choose a final solution afterwards from this set. A solution is Pareto-optimal if it cannot be improved in any objective without getting worse in at least one other objective [1, 2, 3].

We consider MOO of support vector machines (SVMs), which mark the state-of-the-art in machine learning for binary classification in the case of moderate problem dimensionality in terms of the number of training patterns [4, 5, 6]. First, we briefly introduce SVMs from the perspective of MOO. In section 3 we discuss MOO model selection for SVMs. We review model selection criteria, optimization methods with an emphasis on evolutionary MOO, and kernel encodings. Section 4 summarizes results on MOO of SVMs considering model selection criteria based on radius-margin bounds [7]. In section 5 we present a real-world application of the proposed methods: Pedestrian detection for driver assistance systems is a difficult classification task, which can be approached using SVMs [8, 9, 10]. Here fast classifiers with a small false alarm rate are needed. We therefore propose MOO to minimize the false positive rate, the false negative rate, and the complexity of the classifier.

2 Support vector machines

Support vector machines are learning machines based on two key elements: a general purpose learning algorithm and a problem specific kernel that computes the inner product of input data points in a feature space. In this section, we concisely summarize SVMs and illustrate some of the underlying concepts. For an introduction to SVMs we refer to the standard literature [11, 4, 6].

2.1 General SVM learning

We start with a general formulation of binary classification. Let

$$S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)),$$

be the set of training examples, where $y_i \in \{-1, 1\}$ is the label associated with input pattern $\mathbf{x}_i \in X$. The task is to estimate a function f from a given class of functions that correctly classifies unseen examples $(\mathbf{x}, y)$ by the calculation of sign(f($\mathbf{x}$)). The only assumption that is made is that the training data as well as the unseen examples are generated independently by the same, but unknown probability distribution $\mathcal{D}$.

The main idea of SVMs is to map the input vectors to a feature space $\mathcal{H}$, where the transformed data is classified by a linear function f. The transformation $\Phi : X \to \mathcal{H}$ is implicitly done by a kernel $k : X \times X \to \mathbb{R}$, which computes an inner product in the feature space and thereby defines the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$. The kernel matrix $\mathbf{K} = (K_{ij})_{i,j=1}^{\ell}$ has

the entries $K_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. The kernel has to be positive semi-definite, that is, $\mathbf{v}^T \mathbf{K} \mathbf{v} \geq 0$ for all $\mathbf{v} \in \mathbb{R}^\ell$ and all S.

The best function f for classification is the one that minimizes the generalization error, that is, the probability of misclassifying unseen examples $P_{\mathcal{D}}(\text{sign}(f(\mathbf{x})) \neq y)$. Because the example's underlying distribution $\mathcal{D}$ is unknown, a direct minimization is not possible. Thus, upper bounds on the generalization error from statistical learning theory are studied that hold with a probability of $1 - \delta$, $\delta \in (0, 1)$.

We follow the way of [5] for the derivation of SVM learning and give an upper bound that directly incorporates the concepts of margin and slack variables. The margin of an example $(\mathbf{x}_i, y_i)$ with respect to a function $f : X \to \mathbb{R}$ is defined by $y_i f(\mathbf{x}_i)$. If a function f and a desired margin γ are given, the example's slack variable $\xi_i(\gamma, f) = \max(0, \gamma - y_i f(\mathbf{x}_i))$ measures how much the example fails to meet the margin (Figure 1 and 2). It holds [5]:

Theorem 1. *Let $\gamma > 0$ and $f \in \{f_{\mathbf{w}} : X \to \mathbb{R},\ \ f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}),\ \ \|\mathbf{w}\| < 1\}$ a linear function in a kernel-defined RKHS with norm at most 1. Let*

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\}$$

be drawn independently according to a probability distribution $\mathcal{D}$ and fix $\delta \in (0, 1)$. Then with probability at least $1 - \delta$ over samples of size ℓ we have

$$P_{\mathcal{D}}(y \neq \text{sign}(f(\mathbf{x}))) \leq \frac{1}{\ell\gamma} \sum_{i=1}^{\ell} \xi_i + \frac{4}{\ell\gamma} \sqrt{\text{tr}(\mathbf{K})} + 3 \sqrt{\frac{\ln(2/\delta)}{2\ell}} \ ,$$

where $\mathbf{K}$ is the kernel matrix for the training set S.

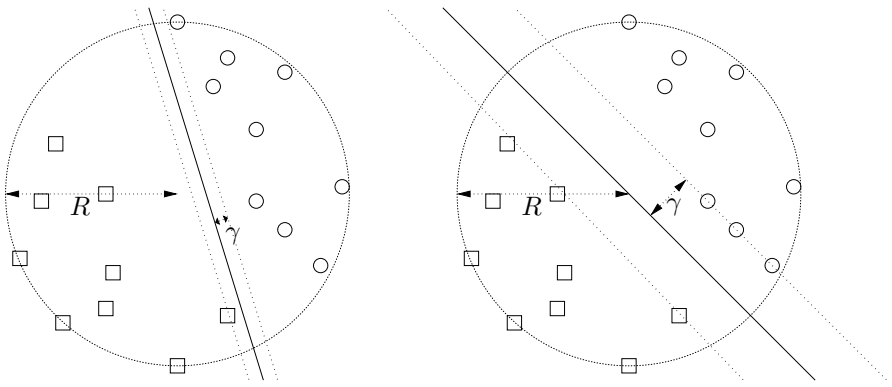

Fig. 1. Two linear decision boundaries separating circles from squares in some feature space. In the right plot, the separating hyperplane maximizes the margin γ, in the left not. The radius of the smallest ball in feature space containing all training examples is denoted by R.

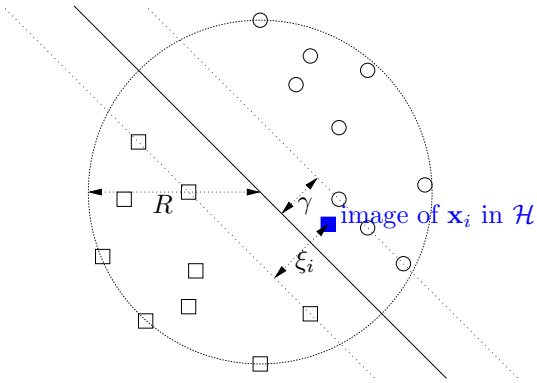

Fig. 2. The concept of slack variables.

The upper bound of Theorem 1 gives a way of controlling the generalization error $P_{\mathcal{D}}(y \neq \text{sign}(f(\mathbf{x})))$. It states that the described learning problem has a multi-objective character with two objectives, namely the margin γ and the sum of the slack variables $\sum_{i=1}^{\ell} \xi_i$.

This motivates the following definition of SVM learning[3]:

$$\mathcal{P}_{\text{SVM}} = \begin{cases} \max & \gamma \\ \min & \sum_{i=1}^{\ell} \xi_i \\ \text{subject to} & y_i((\mathbf{w} \cdot \Phi(\mathbf{x}_i)) + b) \geq \gamma - \xi_i \ , \\ & \xi_i \geq 0, \ \ i = 1, \ldots, \ell \text{ and } \|\mathbf{w}\|^2 = 1 \ . \end{cases}$$

In this formulation γ represents the *geometric margin* due to the fixation of $\|\mathbf{w}\|^2 = 1$. For the solution of $\mathcal{P}_{\text{SVM}}$ all training patterns $(\mathbf{x}_i, y_i)$ with $\xi_i = 0$ have a distance of at least γ to the hyperplane.

The more traditional formulation of SVM learning that will be used throughout this chapter is slightly different. It is a scaled version of $\mathcal{P}_{\text{SVM}}$, but finally provides the same classifier:

$$\mathcal{P}'_{\text{SVM}} = \begin{cases} \min & (\mathbf{w} \cdot \mathbf{w}) \\ \min & \sum_{i=1}^{\ell} \xi_i \\ \text{subject to} & y_i((\mathbf{w} \cdot \Phi(\mathbf{x}_i)) + b) \geq 1 - \xi_i \ , \\ & \xi_i \geq 0, \ \ i = 1, \ldots, \ell \ . \end{cases} \tag{1}$$

There are more possible formulations of SVM learning. For example, when considering the kernel as part of the SVM learning process, as it is done in [12], it becomes necessary to incorporate the term $\text{tr}(\mathbf{K})$ of Theorem 1 into the optimization problem.

[3] We do not take into account that the bound of Theorem 1 has to be adapted in the case $b \neq 0$.

2.2 Classic C-SVM learning

Until now we have only considered multi-objective formulations of SVM learning. In order to obtain the classic single-objective C-SVM formulation the weighted sum method is applied to (1). The factor C determines the trade-off between the margin γ and the sum of the slack variables $\sum_{i=1}^{\ell} \xi_i$:

$$\mathcal{P}_{C\text{-SVM}} = \begin{cases} \min & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C \sum_{i=1}^{\ell} \xi_i \\ \text{subject to} & y_i((\mathbf{w} \cdot \Phi(\mathbf{x}_i)) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \;\; i = 1, \ldots, \ell \end{cases} .$$

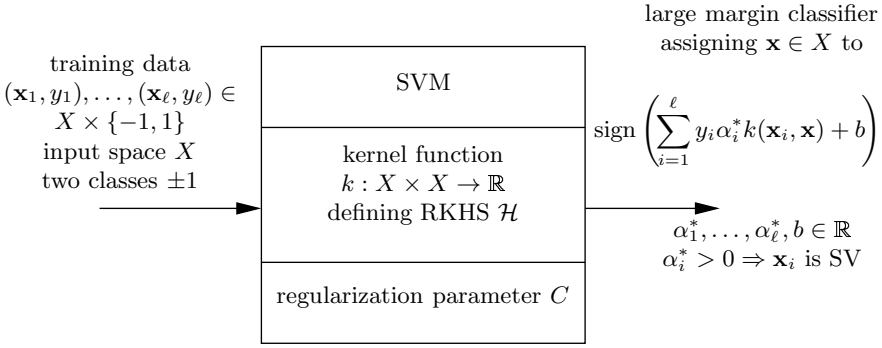

Fig. 3. Classification by a soft margin SVM. The learning algorithm is fully specified by the kernel function k and the regularization parameter C. Given training data, it generates the coefficients of a decision function.

This optimization problem $\mathcal{P}_{C\text{-SVM}}$ defines the soft margin L_1-SVM schematically shown in Figure 3. It can be solved by Lagrangian methods. The resulting classification function becomes

$$\text{sign}(f(\mathbf{x})) \quad \text{with} \quad f(\mathbf{x}) = \sum_{i=1}^{\ell} y_i \alpha_i^* k(\mathbf{x}_i, \mathbf{x}) + b \;\; .$$

The coefficients α_i^* are the solution of the following quadratic optimization problem

$$\begin{aligned} \text{maximize} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} y_i y_j \alpha_i, \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{\ell} \alpha_i y_i = 0 \;\; , \\ & 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, \ell \;\; . \end{aligned} \tag{2}$$

The optimal value for b can then be computed based on the solution $\boldsymbol{\alpha}^*$. The vectors $\mathbf{x}_i$ with $\alpha_i^* > 0$ are called support vectors. The number of support vectors is denoted by $\#\,\mathrm{SV}$. The regularization parameter C controls the trade-off between maximizing the margin

$$\gamma^* = \left(\sum_{i,j=1}^{\ell} y_i y_j \alpha_i^* \alpha_j^* k(\mathbf{x}_i, \mathbf{x}_j) \right)^{-1/2}$$

and minimizing the L_1-norm of the final margin slack vector $\boldsymbol{\xi}^*$ of the training data, where

$$\xi_i^* = \max \left(0, 1 - y_i \left(\sum_{j=1}^{\ell} y_j \alpha_j^* k(\mathbf{x}_j, \mathbf{x}_j) + b \right) \right) \ .$$

In the following we give an extension of the classic C-SVM. It is especially important for practical applications, where the case of highly unbalanced data appears very frequently. To realize a different weighting for wrongly classified positive and negative training examples different cost-factors C_+ and C_- are introduced [13] that change the optimization problem $\mathcal{P}_{C\text{-SVM}}$ to

$$\mathcal{P}_{\tilde{C}\text{-SVM}} = \begin{cases} \min & \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) + C_+ \sum_{i \in I^+} \xi_i + C_- \sum_{i \in I^-} \xi_i \\ \text{subject to} & y_i((\mathbf{w} \cdot \Phi(\mathbf{x}_i)) + b) \geq 1 - \xi_i \ , \\ & \xi_i \geq 0, \ \ i = 1, \ldots, \ell \ , \end{cases}$$

where $I^+ = \{i \in \{1, \ldots, \ell\} \mid \ y_i = 1\}$ and $I^- = \{i \in \{1, \ldots, \ell\} \mid \ y_i = -1\}$. The quadratic optimization problem remains unchanged, except for constraint (2) that has be adapted to

$$0 \leq \alpha_i \leq C_- \, , \ \ i \in I^- \ ,$$

$$0 \leq \alpha_j \leq C_+ \, , \ \ j \in I^+ \ .$$

3 Model selection for SVMs

So far we have considered the inherent multi-objective nature of SVM training. For the remainder of this chapter, we focus on a different multi-objective design problem in the context of C-SVMs. We consider model selection of SVMs, subsuming hyperparameter adaptation and feature selection with respect to different model selection criteria, which are discussed in this section.

Choosing the right kernel for an SVM is crucial for its training accuracy and generalization capabilities as well as the complexity of the resulting classifier. When a parameterized family of kernel functions is considered, kernel

adaptation reduces to finding an appropriate parameter vector. These parameters together with the regularization parameter C are called hyperparameters of the SVM.

In the following, we first discuss optimization methods used for SVM model selection with an emphasis on evolutionary multi-objective optimization. Then different model selection criteria are briefly reviewed. Section 3.3 deals with appropriate encodings for Gaussian kernels and for feature selection.

3.1 Optimization methods for model selection

In practice, the standard method to determine the hyperparameters is grid-search. In simple grid-search the hyperparameters are varied with a fixed step-size through a wide range of values and the performance of every combination is measured. Because of its computational complexity, grid-search is only suitable for the adjustment of very few parameters. Further, the choice of the discretization of the search space may be crucial.

Perhaps the most elaborate techniques for choosing hyperparameters are gradient-based approaches [14, 15, 16, 17, 18]. When applicable, these methods are highly efficient. However, they have some drawbacks and limitations. The most important one is that the score function for assessing the performance of the hyperparameters (or at least an accurate approximation of this function) has to be differentiable with respect to all hyperparameters, which excludes reasonable measures such as the number of support vectors. In some approaches, the computation of the gradient is only exact in the hard-margin case (i.e., for separable data / L_2-SVMs) when the model is consistent with the training data. Further, as the objective functions are indeed multi-modal, the performance of gradient-based heuristics may strongly depend on the initialization—the algorithms are prone to getting stuck in sub-optimal local optima. Evolutionary methods partly overcome these problems.

Evolutionary algorithms

Evolutionary algorithms (EAs) are a class of iterative, direct, randomized global optimization techniques based on principles of neo-Darwinian evolution theory. In canonical EAs, a set of individuals forming the parent population is maintained, where each individual has a genotype that encodes a candidate solution for the optimization problem at hand. 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, new individuals, the offspring, are generated by partially stochastic variations of parent individuals. After the fitness of each offspring has been computed, a selection mechanism that prefers individuals with better fitness chooses the new parent population from the current parents and the offspring. This loop of variation and selection is repeated until a termination criterion is met.

In [19, 20], single-objective evolution strategies were proposed for adapting SVM hyperparameters. A single-objective genetic algorithm for SVM feature selection (see below) was used in [21, 22, 23, 24], where in [22] additionally the (discretized) regularization parameter was adapted.

Multi-objective optimization

Training accuracy, generalization capability, and complexity of the SVM (measured by the number of support vectors) are multiple, probably conflicting objectives. Therefore, it can be beneficial to treat model selection as a multi-objective optimization (MOO) problem.

Consider an optimization problem with M objectives $f_1, \ldots, f_M : X \to \mathbb{R}$ to be minimized. The elements of X can be partially ordered using the concept of Pareto dominance. A solution $\mathbf{x} \in X$ dominates a solution $\mathbf{x}'$ and we write $\mathbf{x} \prec \mathbf{x}'$ if and only if $\exists m \in \{1, \ldots, M\} : f_m(\mathbf{x}) < f_m(\mathbf{x}')$ and $\nexists m \in \{1, \ldots, M\} : f_m(\mathbf{x}) > f_m(\mathbf{x}')$. The elements of the (Pareto) set $\{\mathbf{x} \,|\, \nexists \mathbf{x}' \in X : \mathbf{x}' \prec \mathbf{x}\}$ are called Pareto-optimal. Without any further information, no Pareto-optimal solution can be said to be superior to another element of the Pareto set. The goal of MOO is to find in a single trial a diverse set of Pareto-optimal solutions, which provide insights into the trade-offs between the objectives. When approaching a MOO problem by linearly aggregating all objectives into a scalar function, each weighting of the objectives yields only a limited subset of Pareto-optimal solutions. That is, various trials with different aggregations become necessary—but when the Pareto front (the image of the Pareto set in the m-dimensional objective space) is not convex, even this inefficient procedure does not help (cf. [2, 3]).

Evolutionary multi-objective algorithms have become the method of choice for MOO [1, 2]. Applications of evolutionary MOO to model selection for neural networks can be found in [25, 26, 27, 28], for SVMs in [7, 29, 30].

3.2 Model selection criteria

In the following, we list performance indices that have been considered for SVM model selection. They can be used alone or in linear combination for single-objective optimization. In MOO a subset of these criteria can be used as different objectives.

Accuracy on sample data

The most straightforward way to evaluate a model is to consider its classification performance on sample data. One can always compute the empirical risk given by the error on the training data. To estimate the generalization performance of an SVM, one monitors its accuracy on data not used for training. In the simplest case, the available data is split into a training and validation

set, the first one is used for building the SVM and the second for assessing the performance of the classifier.

In L-fold cross-validation (CV) the available data is partitioned into L disjoint sets $D_1, \ldots, D_L$ of (approximately) equal size. For given hyperparameters, the SVM is trained L times. In the ith iteration, all data but the patterns in D_i are used to train the SVM and afterwards the performance on the ith validation data set D_i is determined. At last, the errors observed on the L validation data sets are averaged yielding the L-fold CV error. In addition, the average empirical risk observed in the L iterations can be computed, a quantity we call L-fold CV training error. The ℓ-fold CV (training) error is called the leave-one-out (training) error. The L-fold CV error is an unbiased estimate of the expected generalization error of the SVM trained with $\lfloor \ell - \ell/L \rfloor$ i.i.d. patterns. Although the bias is low, the variance may not be, in particular for large L. Therefore, and for reasons of computational complexity, moderate choices of L (e.g., 5 or 10) are usually preferred [31].

It can be reasonable to split the classification performance into false negative and false positive rate and consider sensitivity and specificity as two separate objectives of different importance. This topic is discussed in detail in Section 5.

Number of input features

Often the input space X can be decomposed into $X = X_1 \times \cdots \times X_m$. The goal of feature selection is then to determine a subset of indices (feature dimensions) $\{i_1, \ldots, i_{m'}\} \subset \{1, \ldots, m\}$ that yields classifiers with good performance when trained on the reduced input space $X' = X_{i_1} \times \cdots \times X_{i_{m'}}$. By detecting a set of highly discriminative features and ignoring non-discriminative, redundant, or even deteriorating feature dimensions, the SVM may give better classification performance than when trained on the complete space X. By considering only a subset of feature dimensions, the computational complexity of the resulting classifier decreases. Therefore reducing the number of feature dimensions is a common objective.

Feature selection for SVMs is often done using single-objective [21, 22, 23, 24] or multi-objective [29, 30] evolutionary computing. For example, in [30] evolutionary MOO of SVMs was used to design classifiers for protein fold prediction. The three objective functions to be minimized were the number of features, the CV error, and the CV training error. The features were selected out of 125 protein properties such as the frequencies of the amino acids, polarity, and van der Waals volume. In another bioinformatics scenario [29] dealing with the classification of gene expression data using different types of SVMs, the subset of genes, the leave-one-out error, and the leave-one-out training error were minimized using evolutionary MOO.

Modified radius-margin bounds

Bounds on the generalization error derived using statistical learning theory (see Section 2.1) can be (ab)used as criteria for model selection.[4] In the following, we consider radius-margin bounds for L_1-SVMs as used for example in [7] and Section 4.

Let R denote the radius of the smallest ball in feature space containing all ℓ training examples given by

$$R = \sqrt{\sum_{i=1}^{\ell} \beta_i^* K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{\ell} \beta_i^* \beta_j^* K(\mathbf{x}_i, \mathbf{x}_j)} \ ,$$

where $\boldsymbol{\beta}^*$ is the solution vector of the quadratic optimization problem

$$\begin{aligned} \underset{\boldsymbol{\beta}}{\text{maximize}} \quad & \sum_{i=1}^{\ell} \beta_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{\ell} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{\ell} \beta_i = 1 \\ & \beta_i \geq 0 \quad , \quad i = 1, \ldots, \ell \ , \end{aligned}$$

see [32]. The modified radius-margin bound

$$T_{\mathrm{DM}} = (2R)^2 \sum_{i=1}^{\ell} \alpha_i^* + \sum_{i=1}^{\ell} \xi_i^* \ ,$$

was considered for model selection of L_1-SVMs in [33]. In practice, this expression did not lead to satisfactory results [15, 33]. Therefore, in [15] it was suggested to use

$$T_{\mathrm{RM}} = R^2 \sum_{i=1}^{\ell} \alpha_i^* + \sum_{i=1}^{\ell} \xi_i^* \ ,$$

based on heuristic considerations and it was shown empirically that T_{RM} leads to better models than T_{DM}.[5] Both criteria can be viewed as two different aggregations of the following two objectives

$$f_1 = R^2 \sum_{i=1}^{\ell} \alpha_i^* \text{ and } f_2 = \sum_{i=1}^{\ell} \xi_i^* \tag{3}$$

[4] When used for model selection in the described way, the assumptions of the underlying theorems from statistical learning theory are violated and the term "bound" is misleading.

[5] Also for L_2-SVMs it was shown empirically that theoretically better founded weightings of such objectives (e.g., corresponding to tighter bounds) need not correspond to better model selection criteria [15].

penalizing model complexity and training errors, respectively. For example, a highly complex SVM classifier that very accurately fits the training data has high f_1 and small f_2.

Number of support vectors

There are good reasons to prefer SVMs with few support vectors (SVs). In the hard-margin case, the number of SVs (#SV) is an upper bound on the expected number of errors made by the leave-one-out procedure (e.g., see [14, 6]). Further, the space and time complexity of the SVM classifier scales with the number of SVs.

For example, in [7] the number of SVs was optimized in combination with the empirical risk, see also Section 4.

3.3 Encoding in evolutionary model selection

For feature selection a binary encoding is appropriate. Here, the genotypes are n-dimensional bit vectors $(b_1, \ldots, b_n)^T = \{0, 1\}^n$, indicating that the ith feature dimension is used or not depending on $b_i = 1$ or $b_i = 0$, respectively [29, 30].

When a parameterized family of kernel functions is considered, the kernel parameters can be encoded more or less directly. In the following, we focus on the encoding of Gaussian kernels.

The most frequently used kernels are Gaussian functions. General Gaussian kernels have the form

$$k_{\mathbf{A}}(\mathbf{x}, \mathbf{z}) = \exp\left(-(\mathbf{x} - \mathbf{z})^T \mathbf{A}(\mathbf{x} - \mathbf{z})\right)$$

for $\mathbf{x}, \mathbf{z} \in \mathbb{R}^n$ and $\mathbf{A} \in M$, where $M := \{\mathbf{B} \in \mathbb{R}^{n \times n} \,|\, \forall x \neq 0 : x^T \mathbf{B} x > 0 \wedge \mathbf{B} = \mathbf{B}^T\}$ is the set of positive definite symmetric $n \times n$ matrices.

When adapting Gaussian kernels, the questions of how to ensure that the optimization algorithm only generates positive definite matrices arises. This can be realized by an appropriate parameterization of $\mathbf{A}$. Often the search is restricted to $k_{\gamma \mathbf{I}}$, where $\mathbf{I}$ is the unit matrix and $\gamma > 0$ is the only adjustable parameter. However, allowing more flexibility has proven to be beneficial (e.g., see [14, 19, 16]). It is straightforward to allow for independent scaling factors weighting the input components and consider $k_{\mathbf{D}}$, where $\mathbf{D}$ is a diagonal matrix with arbitrary positive entries. This parameterization is used in most of the experiments described in Sections 4 and 5. However, only by dropping the restriction to diagonal matrices one can achieve invariance against linear transformations of the input space. To allow for arbitrary covariance matrices for the Gaussian kernel, that is, for scaling and rotation of the search space, we use a parameterization of M mapping $\mathbb{R}^{n(n+1)/2}$ to M such that all modifications of the parameters by some optimization algorithm always result in feasible kernels. In [19], a parameterization of M is used which was inspired

by the encoding of covariance matrices for mutative self-adaptation in evolution strategies. We make use of the fact that for any symmetric and positive definite $n \times n$ matrix $\mathbf{A}$ there exists an orthogonal $n \times n$ matrix $\mathbf{T}$ and a diagonal $n \times n$ matrix $\mathbf{D}$ with positive entries such that $\mathbf{A} = \mathbf{T}^T\mathbf{D}\mathbf{T}$ and

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

as proven in [34]. The $n \times n$ matrices $\mathbf{R}(\alpha_{i,j})$ are elementary rotation matrices. These are equal to the unit matrix except for $[\mathbf{R}(\alpha_{i,j})]_{ii} = [\mathbf{R}(\alpha_{i,j})]_{jj} = \cos\alpha_{ij}$ and $[\mathbf{R}(\alpha_{i,j})]_{ji} = -[\mathbf{R}(\alpha_{i,j})]_{ij} = \sin\alpha_{ij}$. However, this is not a canonical representation. It is not invariant under reordering the axes of the coordinate system, that is, applying the rotations in a different order (as discussed in the context of evolution strategies in [35]). The natural injective parameterization is to use the exponential map

$$\exp : \mathfrak{m} \to M \ , \qquad \mathbf{A} \mapsto \sum_{i=0}^{\infty} \frac{\mathbf{A}^i}{i!} \ ,$$

where $\mathfrak{m} := \{\mathbf{A} \in \mathbb{R}^{n \times n} \,|\, \mathbf{A} = \mathbf{A}^T\}$ is the vector space of symmetric $n \times n$ matrices, see [16]. However, also the simpler, but non-injective function $\mathfrak{m} \to \overline{M}$ mapping $\mathbf{A} \mapsto \mathbf{A}\mathbf{A}^T$ should work.

4 Experiments on benchmark data

In this section, we summarize results from evolutionary MOO obtained in [7]. In that study, L_1-SVMs with Gaussian kernels were considered and the two objectives given in (3) were optimized.

The evaluation was based on four common medical benchmark datasets *breast-cancer*, *diabetes*, *heart*, and *thyroid* with input dimensions n equal to 9, 8, 13, and 5, and ℓ equal to 200, 468, 170, and 140. The data originally from the UCI Benchmark Repository [36] were preprocessed and partitioned as in [37]. The first of the splits into training and external test set D_{train} and D_{extern} was considered.

Figure 4 shows the results of optimizing $k_{\gamma\mathbf{I}}$ (see Section 3.3) using the objectives (3). For each f_1 value of a solution the corresponding f_2, T_{RM}, T_{DM}, and the percentage of wrongly classified patterns in the test data set $100 \cdot \text{CE}(D_{\text{extern}})$ are given. For *diabetes*, *heart*, and *thyroid*, the solutions lie on typical convex Pareto fronts; in the *breast-cancer* example the convex front looks piecewise linear.

Assuming convergence to the Pareto-optimal set, the results of a single MOO trial are sufficient to determine the outcome of single-objective optimization of any (positive) linear weighting of the objectives. Thus, we can

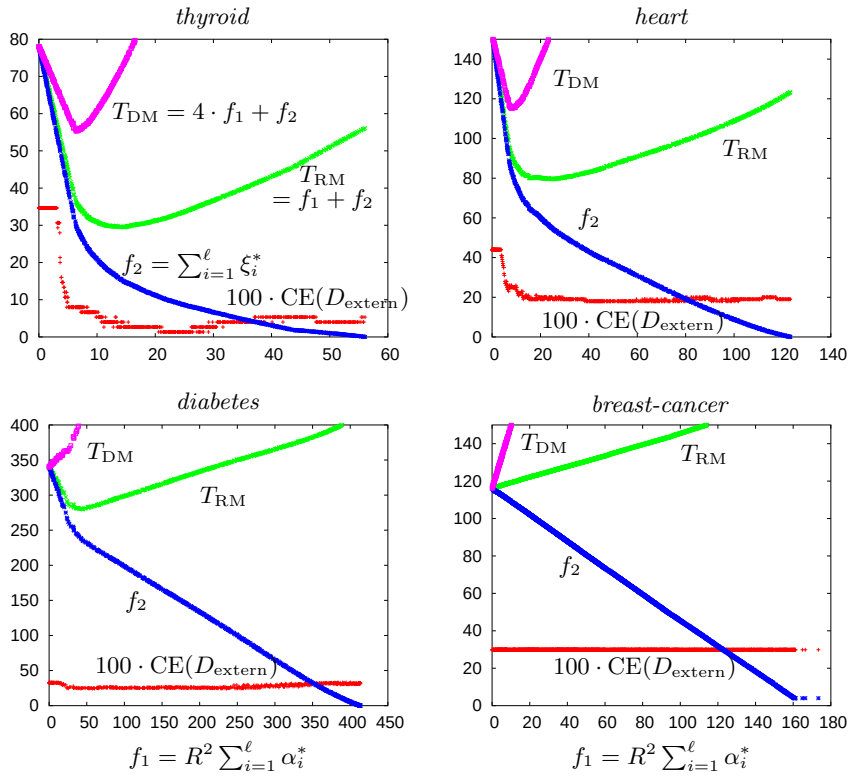

Fig. 4. Pareto fronts (i.e., (f_1, f_2) of non-dominated solutions) after 1500 fitness evaluations, see [7] for details. For every solution the values of T_{RM}, T_{DM}, and $100 \cdot \mathrm{CE}(D_{\mathrm{extern}})$ are plotted against the corresponding f_1 value, where $\mathrm{CE}(D_{\mathrm{extern}})$ is the proportion of wrongly classified patterns in the test data set. Projecting the minimum of T_{RM} (for T_{DM} proceed analogously) along the y-axis on the Pareto front gives the (f_1, f_2) pair suggested by the model selection criterion T_{RM}—this would also be the outcome of single-objective optimization using T_{RM}. Projecting an (f_1, f_2) pair along the y-axis on $100 \cdot \mathrm{CE}(D_{\mathrm{extern}})$ yields the corresponding error on an external test set.

directly determine and compare the solutions that minimizing T_{RM} and T_{DM} would suggest.

The experiments confirm the findings in [15] that the heuristic bound T_{RM} is better suited for model selection than T_{DM}. When looking at $\mathrm{CE}(D_{\mathrm{extern}})$ and the minima of T_{RM} and T_{DM}, we can conclude that T_{DM} puts too much emphasis on the "radius-margin part" yielding worse classification results on the external test set (except for *breast-cancer* where there is no difference on D_{extern}). The *heart* and *thyroid* results suggest that even more weight should

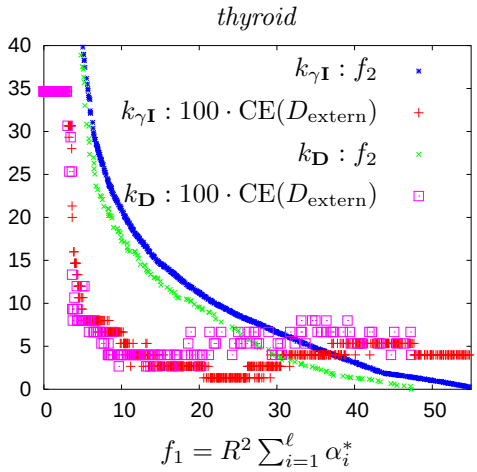

Fig. 5. Pareto fronts after optimizing $k_{\gamma\mathbf{I}}$ and $k_{\mathbf{D}}$ for objectives (3) and *thyroid* data after 1500 fitness evaluations [7]. For both kernel parameterizations, f_2 and $100 \cdot \mathrm{CE}(D_{\mathrm{extern}})$ are plotted against f_1.

be given to the slack variables (i.e., the performance on the training set) than in T_{RM}.

In the MOO approach, degenerated solutions resulting from a not appropriate weighting of objectives (which we indeed observed—without the chance to change the trade-off afterwards—in single-objective optimization of SVMs) become obvious and can be excluded. For example, one would probably not pick the solution suggested by T_{DM} in the *diabetes* benchmark. A typical MOO heuristic is to choose a solution that belongs to the "interesting" part of the Pareto front. In case of a typical convex front, this would be the area of highest "curvature" (the "knee", see Figure 4). In our benchmark problems, this leads to results on a par with T_{RM} and much better than T_{DM} (except for *breast-cancer*, where the test errors of all optimized trade-offs were the same). Therefore, this heuristic combined with T_{RM} (derived from the MOO results) is an alternative for model selection based on modified radius margin bounds.

Adapting the scaling of the kernel (i.e., optimizing $k_{\mathbf{D}}$) sometimes led to better objective values compared to $k_{\gamma\mathbf{I}}$, see Figure 5 for an example, but not necessarily to better generalization performance.

5 Real-world application: Pedestrian detection

In this section, we consider MOO of SVM classifiers for online pedestrian detection in infrared images for driver assistance systems. This is a challenging real-world task with strict real-time constraints requiring highly optimized

classifiers and a considerate adjustment of sensitivity and specificity. Instead of optimizing a single SVM and varying the bias parameter to get a ROC (receiver operating characteristic) curve [8, 38], we apply MOO to decrease the false positive rate, the false negative rate, as well as the number of support vectors. Reducing the latter directly corresponds to decreasing the capacity and the computational complexity of the classifier. We automatically select the kernel parameters, the regularization parameter, and the weighting of positive and negative examples during training. Gaussian kernel functions with individual scaling parameters for each component of the input are adapted. As neither gradient-based optimization methods nor grid-search techniques are applicable, we solve the problem using the real-valued non-dominated sorting genetic algorithm NSGA-II [2, 39].

5.1 Pedestrian detection

Robust object detection systems are a key technology for the next generation of driver assistance systems. They make major contributions to the environment representation of the ego-vehicle, which serves a basis for different high-level driver assistance applications. Besides vehicle detection the early detection of pedestrians is of great interest since it is one important step towards avoiding dangerous situations. In this section we focus on the special case of the detection of pedestrians in a single frame. This is an extremely difficult problem, because of the large variety of human appearances, as pedestrians are standing or walking, carrying bags, wearing hats, etc. Another reason making pedestrian detection very difficult is that pedestrians usually appear in urban environment with complex background (e.g., containing buildings, cars, traffic signs, and traffic lights).

Most of the past work in detecting pedestrians was done using visual cameras. These approaches use a lot of different techniques so we can name only a few. In [40] segmentation was done by means of stereo vision and classification by the use of neural networks. Classification with SVMs that are working on wavelet features was suggested in [41]. A shape-based method for classification was applied in [42]. In [43] a hybrid approach for pedestrian detection was presented, which evaluates the leg-motion and tracks the upper part of the pedestrian.

Recently some pedestrian detection systems have been developed that are working with infrared images, where the color depends on the heat of the object. The advantage of infrared based systems is that they are almost independent on the lighting conditions, so that night-vision is possible. A shape-based method for the classification of pedestrians in infrared images was developed by [44, 45] and an SVM-based one was suggested in [46].

5.2 Pedestrian detection system

In this section we give a description of our pedestrian detection system that is working with infrared images. We keep it rather short because our focus mainly lies on the classification task.

The task of detecting pedestrians is usually divided into two steps, namely the segmentation of candidate regions for pedestrians and the classification of the segmented regions (Figure 6). In our system the segmentation of candidate

Fig. 6. Results of our pedestrian detection system on an infrared image; the left picture shows the result of the segmentation step, which provides candidate regions for pedestrians; the image on the right shows the regions that has been labeled as pedestrian.

regions for pedestrians is based on horizontal gradient information, which is used to find vertical structures in the image. If the candidate region is at least of size 10×20 pixels a feature vector is calculated and classified using an SVM.

The calculation of the feature vectors is based on contour points and the corresponding discretized angles, which are obtained using a Canny filter [47]. To make the approach scaling-invariant we put a 4×8 grid on the candidate region and determine the histograms of eight different angles for each of these fields. In a last step the resulting 256-dimensional feature vector is normalized to the range $[-1, 1]^{256}$.

5.3 Model selection

In practice the common way for assigning a performance to a classifier is to analyze its ROC curve. This analysis visualizes the trade-off between the two partially conflicting objectives false negative and false positive rate and allows for the selection of a problem specific solution. A third objective for SVM model-selection, which is especially important for real-time tasks like pedestrian detection, is the number of support vectors, because it directly determines the computational complexity of the classifier.

We use an EA for the tuning of much more parameters than would be possible with grid-search, thus making a better adaptation to the given problem possible. Concretely we tune the parameters C_+, C_-, and $\mathbf{D}$, that is, independent scaling factors for each component of the feature vector (see Sections 2.2 and 3.3).

For the optimization we generated four datasets D_{train}, D_{val}, D_{test}, and D_{extern}, whose use will become apparent in the discussion of the optimization algorithm. Each of the datasets consists of candidate regions (256-dimensional feature vectors) that are manually labeled pedestrian or non-pedestrian. The candidate regions are obtained by our segmentation algorithm to ensure that the datasets are realistic in that way that all usually appearing critical cases are contained. Furthermore the segmentation algorithm provides much more non-pedestrians than pedestrians and therefore negative and positive examples in the data are highly unbalanced. The datasets are obtained from different image sequences, which have been captured on the same day to ensure similar environmental conditions, but no candidate region from the same sequence is in the same dataset.

For optimization we use the NSGA-II, where the fitness of an individual is determined on dataset D_{val} with an SVM that has been trained on dataset D_{train}. This training is done using the individual's corresponding SVM parameterization. To avoid overfitting we keep an external archive of non-dominated solutions, which have been evaluated on the validation set D_{test} for every individual that has been created by the optimization process. The dataset D_{extern} is used for the final evaluation (cf. [28]).

For the application of the NSGA-II we choose a population size of 50 and create the initial parent population by randomly selecting non-dominated solutions from a 3D-grid-search on the parameters C_+, C_-, and one global scaling factor γ, that is $\mathbf{D} = \gamma\mathbf{I}$. The other parameters of the NSGA-II are chosen like in [39] ($p_c = 0.9$, $p_m = 1/n$, $\eta_c = 20$, $\eta_m = 20$). We carried out 10 optimization trials, each of them lasting for 250 generations.

5.4 Results

In this section we give a short overview about the results of the MOO of the pedestrian detection system.

The progress of one optimization trial is exemplary shown in Figure 7. It illustrates the Pareto-optimal solutions in the objective space that are contained in the external archive after the first and after the 250th generation. The solutions in the archive after the first generation roughly correspond to the solutions that have been found by 3D-grid search. The solutions after the 250th generation have obviously improved and clearly reveal the trade-off between the three objectives, thereby allowing for a problem-specific choice of an SVM.

For assessing the performance of a stochastic optimization algorithm it is not sufficient to evaluate a single optimization trial. A possibility for visual-

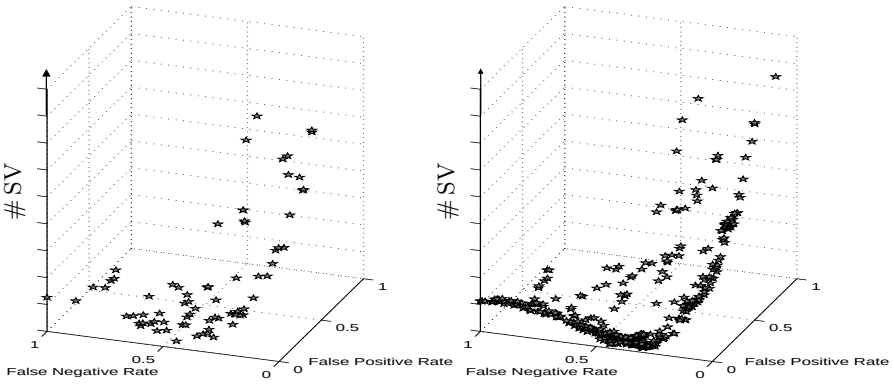

Fig. 7. Pareto-optimal solutions that are contained in the external archive after the first (left plot) and after the 250th generation (right plot).

izing the outcome of a series of optimization trials are the so-called summary attainment surfaces [48] that provide the points in objective space that have been attained in a certain fraction of all trials.

We give the summary attainment curve for the two objectives true positive and false positive rate, which are the objectives of the ROC curve. Figure 8 shows the points that have been attained by all, 50%, and the best of our optimization trials.

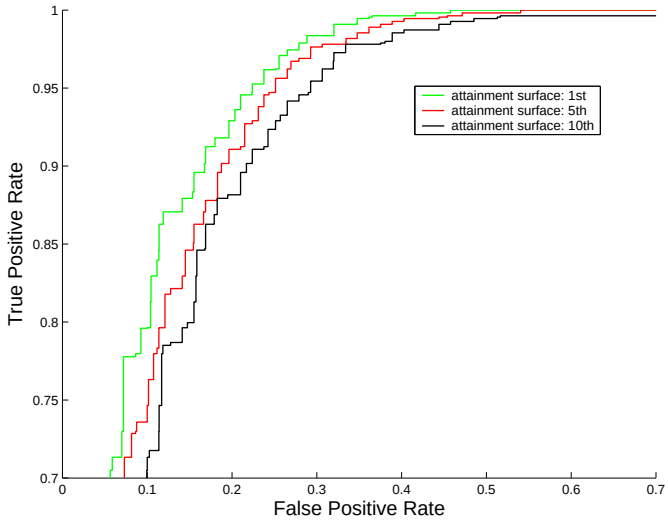

Fig. 8. Summary attainment curves for the objectives true positive rate and false positive rate ("ROC curves").

6 Conclusions

Designing classifiers is a multi-objective optimization (MOO) problem. The application of "true" MOO algorithms allows for visualizing trade-offs, for example between model complexity and learning accuracy or sensitivity and specificity, for guiding the model selection process.

We considered evolutionary MOO of support vector machines (SVMs). This approach can adapt multiple hyperparameters of SVMs based on conflicting, not differentiable criteria.

When optimizing the norm of the slack variables and the radius-margin quotient as two objectives, it turned out that standard MOO heuristics based on the curvature of the Pareto front led to comparable models as corresponding single-objective criteria proposed in the literature. In benchmark problems it appears that the latter should put more emphasis on minimizing the slack variables.

We demonstrated MOO of SVMs for the detection of pedestrians in infrared images for driver assistance systems. Here the three objectives are the false positive rate, the false negative rate, and the number of support vectors. The Pareto front of the first two objectives can be viewed as a ROC curve where each point corresponds to a learning machine optimized for that particular trade-off between sensitivity and specificity. The third objective reduces the model complexity in order to meet real-time constraints.

Acknowledgments

We thank Tobias Glasmachers for proofreading and Aalzen Wiegersma for providing the thoroughly preprocessed pedestrian image data. We acknowledge support from BMW Group Research and Technology.

References

1. Coello Coello, C.A., Van Veldhuizen, D.A., Lamont, G.B.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers (2002)
2. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley (2001)
3. Sawaragi, Y., Nakayama, H., Tanino, T.: Theory of Multiobjective Optimization. Volume 176 of Mathematics in Science and Engineering. Academic Press (1985)
4. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press (2002)
5. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press (2004)
6. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag (1995)

7. Igel, C.: Multi-objective model selection for support vector machines. In Coello Coello, C.A., Zitzler, E., Hernandez Aguirre, A., eds.: Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005). Volume 3410 of LNCS., Springer-Verlag (2005) 534–546
8. Mohan, A., Papageorgiou, C., Poggio, T.: Example-based object detection in images by components. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001) 349–361
9. Oren, M., Papageorgiou, C.P., Sinha, P., Osuna, E., Poggio, T.: Pedestrian detection using wavelet templates. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (1997) 193–199
10. Papageorgiou, C., Poggio, T.: A trainable system for object detection. International Journal of Computer Vision **38** (2000) 15–33
11. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press (2000)
12. Bi, J.: Multi-objective programming in SVMs. In Fawcett, T., Mishra, N., eds.: Machine Learning, Proceedings of the 20th International Conference (ICML 2003), AAAI Press (2003) 35–42
13. Morik, K., Brockhausen, P., Joachims, T.: Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In: Proceedings of the 16th International Conference on Machine Learning, Morgan Kaufmann (1999) 268–277
14. Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S.: Choosing multiple parameters for support vector machines. Machine Learning **46** (2002) 131–159
15. Chung, K.M., Kao, W.C., Sun, C.L., Lin, C.J.: Radius margin bounds for support vector machines with RBF kernel. Neural Computation **15** (2003) 2643–2681
16. Glasmachers, T., Igel, C.: Gradient-based adaptation of general gaussian kernels. Neural Computation **17** (2005) 2099–2105
17. Gold, C., Sollich, P.: Model selection for support vector machine classification. Neurocomputing **55** (2003) 221–249
18. Keerthi, S.S.: Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. IEEE Transactions on Neural Networks **13** (2002) 1225–1229
19. Friedrichs, F., Igel, C.: Evolutionary tuning of multiple SVM parameters. Neurocomputing **64** (2005) 107–117
20. Runarsson, T.P., Sigurdsson, S.: Asynchronous parallel evolutionary model selection for support vector machines. Neural Information Processing – Letters and Reviews **3** (2004) 59–68
21. Eads, D.R., Hill, D., Davis, S., Perkins, S.J., Ma, J., Porter, R.B., Theiler, J.P.: Genetic algorithms and support vector machines for time series classification. In Bosacchi, B., Fogel, D.B., Bezdek, J.C., eds.: Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation V. Volume 4787 of Proceedings of the SPIE. (2002) 74–85
22. Fröhlich, H., Chapelle, O., Schölkopf, B.: Feature selection for support vector machines using genetic algorithms. International Journal on Artificial Intelligence Tools **13** (2004) 791–800
23. Jong, K., Marchiori, E., van der Vaart, A.: Analysis of proteomic pattern data for cancer detection. In Raidl, G.R., Cagnoni, S., Branke, J., Corne, D.W., Drechsler, R., Jin, Y., Johnson, C.G., Machado, P., Marchiori, E., Rothlauf,

F., Smith, G.D., Squillero, G., eds.: Applications of Evolutionary Computing. Volume 3005 of LNCS., Springer-Verlag (2004) 41–51
24. Miller, M.T., Jerebko, A.K., Malley, J.D., Summers, R.M.: Feature selection for computer-aided polyp detection using genetic algorithms. In Clough, A.V., Amini, A.A., eds.: Medical Imaging 2003: Physiology and Function: Methods, Systems, and Applications. Volume 5031 of Proceedings of the SPIE. (2003) 102–110
25. Abbass, H.A.: An evolutionary artificial neural networks approach for breast cancer diagnosis. Artificial Intelligence in Medicine **25** (2002) 265–281
26. Abbass, H.A.: Speeding up backpropagation using multiobjective evolutionary algorithms. Neural Computation **15** (2003) 2705–2726
27. Jin, Y., Okabe, T., Sendhoff, B.: Neural network regularization and ensembling using multi-objective evolutionary algorithms. In: Congress on Evolutionary Computation (CEC'04), IEEE Press (2004) 1–8
28. Wiegand, S., Igel, C., Handmann, U.: Evolutionary multi-objective optimization of neural networks for face detection. International Journal of Computational Intelligence and Applications **4** (2004) 237–253 Special issue on Neurocomputing and Hybrid Methods for Evolving Intelligence.
29. Pang, S., Kasabov, N.: Inductive vs. transductive inference, global vs. local models: SVM, TSVM, and SVMT for gene expression classification problems. In: International Joint Conference on Neual Networks (IJCNN). Volume 2., IEEE Press (2004) 1197–1202
30. Shi, S.Y.M., Suganthan, P.N., Deb, K.: Multi-class protein fold recognition using multi-objective evolutionary algorithms. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. (2004) 61–66
31. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer Series in Statistics. Springer-Verlag (2001)
32. Schölkopf, B., Burges, C.J.C., Vapnik, V.: Extracting support data for a given task. In Fayyad, U.M., Uthurusamy, R., eds.: Proceedings of the First International Conference on Knowledge Discovery & Data Mining, Menlo Park, CA, AAAI Press (1995) 252–257
33. Duan, K., Keerthi, S.S., Poo, A.: Evaluation of simple performance measures for tuning SVM hyperparameters. Neurocomputing **51** (2003) 41–59
34. Rudolph, G.: On correlated mutations in evolution strategies. In Männer, R., Manderick, B., eds.: Parallel Problem Solving from Nature 2 (PPSN II), Elsevier (1992) 105–114
35. Hansen, N.: Invariance, self-adaptation and correlated mutations and evolution strategies. In: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature (PPSN VI). Volume 1917 of LNCS., Springer-Verlag (2000) 355–364
36. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
37. Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for AdaBoost. Machine Learning **42** (2001) 287–320
38. Papageorgiou, C.P.: A trainable system for object detection in images and video sequences. Technical Report AITR-1685, Massachusetts Institute of Technology, Artificial Intelligene Laboratory (2000)

39. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6** (2002) 182–197
40. Zhao, L., Thorpe, C.: Stereo- and neural network-based pedestrian detection. In: Proceedings of the IEEE International Conference on Intelligent Transportation Systems'99. (1999) 298–303
41. Papageorgiou, C., Evgeniou, T., Poggio, T.: A trainable pedestrian detection system. In: Proceedings of the IEEE International Conference on Intelligent Vehicles Symposium 1998. (1998) 241–246
42. Bertozzi, M., Broggi, A., Fascioli, A., Sechi, M.: Shape-based pedestrian detection. In: Proceedings of the IEEE Intelligent Vehicles Symposium 2000. (2000) 215–220
43. Curio, C., Edelbrunner, J., Kalinke, T., Tzomakas, C., von Seelen, W.: Walking pedestrian recognition. IEEE Transactions on Intelligent Transportation Systems **1** (2000) 155–163
44. Bertozzi, M., Broggi, A., Carletti, M., Fascioli, A., Graf, T., Grisleri, P., Meinecke, M.: IR pedestrian detection for advanced driver assistance systems. In: Proceedings of the 25th Pattern Recognition Symposium. Volume 2781 of LNCS., Springer-Verlag (2003) 582–590
45. Bertozzi, M., Broggi, A., Graf, T., Grisleri, P., Meinecke, M.: Pedestrian detection in infrared images. In: Proceedings of the IEEE Intelligent Vehicles Symposium 2003. (2003) 662–667
46. Xu, F., Liu, X., Fujimura, K.: Pedestrian detection and tracking with night vision. IEEE Transactions on Intelligent Transportation Systems **6** (2005) 63–71
47. Canny, J.: A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **8** (1986) 679–698
48. Fonseca, C.M., Knowles, J.D., Thiele, L., Zitzler, E.: A tutorial on the performance assessment of stochastic multiobjective optimizers. Presented at the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005) (2005)