Project number IST-25582

CGL

Computational Geometric Learning

Basis Expansions for Support Vector Machines

STREP

Information Society Technologies

Period covered:	November 1, 2010–October 31, 2011
Date of preparation:	October 24, 2012
Date of revision:	October 24, 2012
Start date of project:	November 1, 2010
Duration:	3 years
Project coordinator name:	Joachim Giesen (FSU)
Project coordinator organisation:	Friedrich-Schiller-Universität Jena Jena, Germany

Basis Expansions for Support Vector Machines

Joachim Giesen* Sören Laue† Jens K. Mueller‡

October 24, 2012

Abstract

Recently, it has been pointed out by Chapelle that there is no reason not to consider training Support Vector Machines (SVMs) in the primal, although training SVMs is still almost exclusively introduced using the dual formulation. Here we explore the use of kernels in the primal formulation of SVMs on some set Ω. Working in the primal shifts the focus from kernels on Ω to orthonormal bases of Hilbert spaces of functions on Ω. Such basis functions can be derived from a kernel on Ω, but in general the basis function view offers more flexibility. We discuss two examples of this added flexibility, namely non-uniformity and incremental construction of basis functions.

1 Introduction

Training large scale primal Support Vector Machines (SVMs) has been studied by Keerthi and DeCoste [9], and recently Chapelle [4] has pointed out that also non-linear primal SVMs can be trained efficiently. The focus in both works has been on the optimization algorithms. Here we want to focus on the modeling aspects when applying the kernel trick in the primal SVM formulation.

SVMs are geometric in nature, i.e., they build on a Euclidean or Hilbert space structure which allows to measure distances and angles. Let $x_1, \ldots, x_n$ be data points in some real (typically finite dimensional) Hilbert space $\mathcal{H}$ together with binary labels $y_1, \ldots, y_n \in \{-1, 1\}$. Let $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ be the scalar product on $\mathcal{H}$. The primal soft margin SVM is the following convex quadratic optimization problem

primal **SVM** :	$\min_{w \in \mathcal{H}, \xi \in \mathbb{R}^n, b \in \mathbb{R}}$	$\frac{1}{2}\|w\|_{\mathcal{H}}^2 + c \sum_{i=1}^n \xi_i$	
	s.t.	$y_i \left(\langle w, x_i \rangle_{\mathcal{H}} + b \right) \geq 1 - \xi_i$ and $\xi \geq 0$,	

where $\| \cdot \|_{\mathcal{H}}$ is the norm on $\mathcal{H}$ derived from the scalar product on $\mathcal{H}$, and the trade-off between the regularization term $\frac{1}{2}\|w\|^2$ and the loss term $\sum_{i=1}^n \xi_i$ is parameterized by the regularization parameter c. The classifier derived from an optimal solution (w, b) of the SVM is given as

$$h : \mathcal{H} \to \{-1, 1\}, x \mapsto \operatorname{sgn}\big(\langle w, x \rangle_{\mathcal{H}} + b \big).$$

The optimal primal solution w can be computed from the optimal solution α of the Lagrangian dual of the SVM

*Friedrich-Schiller-Universität Jena
†Friedrich-Schiller-Universität Jena
‡Friedrich-Schiller-Universität Jena

dual **SVM** :	$\max_{\alpha\in\mathbb{R}^n}$	$-\frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle_{\mathcal{H}} + \sum_{i=1}^n \alpha_i$
	s.t.	$\sum_{i=1}^n \alpha_i y_i = 0$ and $0 \leq \alpha \leq c$

as $w = \sum_{i=1}^n \alpha_i y_i x_i$, i.e., the optimal w is a linear combination of the data points. The latter is also known as the "representer theorem" [10]. The dual SVM depends only on the scalar products of the data points. This is where the kernel trick comes into the game: given data points $x_1, \ldots, x_n$ from some set Ω together with binary labels $y_1, \ldots, y_n \in \{-1, 1\}$, and a positive kernel k on Ω, we can compute a classifier on Ω from an optimal solution to the convex quadratic optimization problem

kernelized dual **SVM** :	$\max_{\alpha\in\mathbb{R}^n}$	$-\frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j k(x_i, x_j) + \sum_{i=1}^n \alpha_i$
	s.t.	$\sum_{i=1}^n \alpha_i y_i = 0$ and $0 \leq \alpha \leq c$.

The resulting classifier is given as

$$h : \Omega \to \{-1, 1\}, x \mapsto \operatorname{sgn}\left(\left(\sum_{i=1}^n \alpha_i y_i k(x_i, x)\right) + b\right).$$

The purpose of this paper is to explore the use of positive kernels on Ω or more generally orthogonal basis functions to compute classifiers on Ω from labelled data points through primal SVMs.

In the following we will first derive a non-linear formulation of the primal SVM and then highlight some of the added flexibility offered by the alternative basis functions view. Our exposition is mostly of a pedagogical nature promoting an alternative way of teaching non-linearity and SVMs, but we also provide some practical evidence that the basis function view can be beneficial.

2 Applying the kernel trick in the primal

Chapelle's derivation of the kernel trick in the primal requires a differentiable loss function in the unconstrained primal SVM formulation. Here we give an alternative derivation that also applies to the standard primal soft margin SVM whose loss function is not differentiable in the unconstrained formulation.

Positive kernels, i.e., bivariate functions $k : \Omega \times \Omega \to \mathbb{R}$ such that there exists a feature map $\phi : \Omega \to \mathcal{H}_k$ into the real reproducing Hilbert space $\mathcal{H}_k$ with $\langle \phi(x), \phi(y)\rangle_k = k(x, y)$, where $\langle \cdot, \cdot\rangle_k$ is the scalar product on $\mathcal{H}_k$, allow using linear machine learning techniques on general sets Ω, i.e., also spaces without any other structure than the kernel. This is known as the "kernel trick" [2].

Suppose we are given labels $y_1, \ldots, y_n \in \{-1, 1\}$ at data points $x_1, \ldots, x_n$ in some set Ω equipped with a positive kernel k. Instead of the data points we can consider the (dual) elements (functions) k_i in the real reproducing kernel Hilbert space $\mathcal{H}_k$,

$$k_i : \Omega \to \mathbb{R}, x \mapsto k(x_i, x).$$

Let $\mathcal{H}$ be the subspace of $\mathcal{H}_k$ spanned by the functions k_i. The induced scalar product $\langle \cdot, \cdot\rangle_k$ on $\mathcal{H}$ is given by

$$\langle w, v\rangle_k = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \beta_j k(x_i, x_j)$$

if $w = \sum_{i=1}^n \alpha_i k_i \in \mathcal{H}$ and $v = \sum_{i=1}^n \beta_i k_i \in \mathcal{H}$, and thus the squared $\mathcal{H}$-norm of w is

$$\|w\|_k^2 = \langle w, w \rangle_k = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j k(x_i, x_j).$$

Especially, it is $\|k_i\|_k^2 = k(x_i, x_i)$.

Now we want to move from the basis $\{k_i\}$ of $\mathcal{H}$ to an orthonormal basis $\{e_i\}$ of $\mathcal{H}$. This can be done by using either the Gram-Schmidt orthonormalization scheme

$$\begin{aligned} e_1 &= \frac{k_1}{\sqrt{k(x_1, x_1)}} \\ e_i &= \frac{k_i - \sum_{j=1}^{i-1} \langle k_i, e_j \rangle_k e_j}{\| k_i - \sum_{j=1}^{i-1} \langle k_i, e_j \rangle_k e_j \|_k}, \quad i = 2, \ldots, n \end{aligned}$$

or by a singular value decomposition of the positive definite $(n \times n)$-kernel-matrix $K = \big(k(x_i, x_j)\big)$. Since K is positive definite there exists an orthogonal $(n \times n)$-matrix $U = (u_{ij})$ such that $U^t K U = D(\lambda_1, \ldots, \lambda_n)$ where D is a diagonal matrix. From this we obtain an orthonormal basis of $\mathcal{H}$ as

$$e_i = \frac{1}{\sqrt{\lambda_i}} \sum_{j=1}^n u_{ij} k_j, i = 1, \ldots, n.$$

Expanding $w \in \mathcal{H}$ in a basis $\{e_i\}$ as $w = \sum_{i=1}^n \hat{w}_i e_i$ with coefficients $\hat{w}_i \in \mathbb{R}$ gives the following simple equation for the squared $\mathcal{H}$-norm of w,

$$\|w\|_k^2 = \sum_{i=1}^n \hat{w}_i^2 = \|\hat{w}\|_2^2,$$

i.e., $\|w\|_k$ is the squared Euclidean norm of the coefficient vector $\hat{w} = (\hat{w}_1, \ldots, \hat{w}_n)$.

Assume now that at the data points x_i we have observed labels $y_i \in \{-1, 1\}$, and let w be the optimal solution of the corresponding primal SVM. By the representer theorem [10] the optimal solution w is contained in $\mathcal{H}$. If we expand w in the basis $\{e_i\}$, i.e., $w = \sum_{i=1}^n \hat{w}_i e_i$ for optimal coefficients $\hat{w}_i \in \mathbb{R}$, then the standard SVM reads as follows

kernelized primal **SVM** :	$\min_{\hat{w} \in \mathbb{R}^n, \xi \in \mathbb{R}^n, b \in \mathbb{R}}$	$\frac{1}{2}\|\hat{w}\|_2^2 + c \sum_{i=1}^n \xi_i$
	s.t.	$y_i \left(\langle \hat{w}, \hat{x}_i \rangle + b \right) \geq 1 - \xi_i$ and $\xi \geq 0$,

where $\langle \cdot, \cdot \rangle$ is the standard scalar product on $\mathbb{R}^n$, and

$$\hat{x}_i = (\hat{x}_{i1}, \ldots, \hat{x}_{in}) = \big(\langle k_i, e_1 \rangle_k, \ldots, \langle k_i, e_n \rangle_k \big) = \big(e_1(x_i), \ldots, e_n(x_i) \big),$$

where all the $\hat{x}_{ij}$ can be expressed through elements of the kernel matrix $K = \big(k(x_i, x_j)\big)$. Note, that as an optimization problem the kernelized primal SVM is just a linear primal SVM, only the data points $\hat{x}_i$ have to be computed through the kernel function. Hence, any (dedicated) solver for *linear* primal SVMs can also be used to optimize kernelized primal SVMs, for example LIBLINEAR [5], PEGASOS [11], or the solver by Keerthi and DeCoste [9].

3 More general basis functions

Basis functions can be but need not be derived from kernels. Instead of a positive kernel and the associated reproducing kernel Hilbert space we can also consider some Hilbert space $\mathcal{H} \subseteq \mathcal{F}(\Omega)$, where $\mathcal{F}(\Omega)$ is the space of all real valued functions on Ω. Assume that $\mathcal{H}$ is finite and spanned by the orthonormal basis functions $e_i : \Omega \to \mathbb{R}, i = 1, \ldots, m$.

Given data point points $x_1, \ldots, x_n \in \Omega$ together with labels $y_1, \ldots, y_n \in \{-1, 1\}$, we can compute a classifier

$$h : \mathcal{H} \to \{-1, 1\}, x \mapsto \operatorname{sgn}\left(\Big(\sum_{i=1}^{m} w_i y_i e_i(x) \Big) + b \right)$$

from an optimal solution w, b of the following support vector machine

non-linear primal **SVM** :	$\min_{w \in \mathbb{R}^m, \xi \in \mathbb{R}^n, b \in \mathbb{R}}$	$\frac{1}{2}\|w\|_2^2 + c \sum_{i=1}^{n} \xi_i$	
	s.t.	$y_i \left(\langle w, \hat{x}_i \rangle + b \right) \geq 1 - \xi_i$	and $\xi \geq 0$,

where

$$\hat{x}_i = (\hat{x}_{i1}, \ldots, \hat{x}_{im}) = \big(e_1(x_i), \ldots, e_m(x_i) \big).$$

Note that it is not required that $m = n$, i.e., we can also consider less or more basis functions than data points. This is not possible in the standard dual formulation of an SVM and thus can be considered a conceptual advantage of the primal formulation. Another advantage of using basis functions is that they do not need to be centered at the data points. For example, one could use an unsupervised method like a clustering algorithm and center the basis functions at the cluster centers.

3.1 Non-uniformity

Using basis functions instead of kernels offers more flexibility since the basis functions need not be "uniform" in hyper-parameters in contrast to the basis functions $k_i : \Omega \to \mathbb{R}, x \mapsto k(x_i, x)$ derived from a kernel k on Ω.

Often the kernel hyper-parameter can be regarded as some sort of scale selection, i.e., one is analyzing the data at the scale given by the hyper-parameter. If there exist different scales in a data set, then a single hyper-parameter can only comprise between these scales, whereas basis functions that are not uniform in the hyper-parameters allow to adapt to the different scales.

We illustrate the general idea by a simple example for $\Omega = \mathbb{R}^d$. Consider the subspace of the Hilbert space of all square integrable functions $L_2(\mathbb{R}^d) \subset \mathcal{F}(\mathbb{R}^d)$, where the scalar product of two functions $f, g \in L_2(\mathbb{R}^d)$ is given as

$$\langle f, g \rangle = \int_{\mathbb{R}^d} f g \, d\mu,$$

spanned by the d-dimensional normalized Gaussians

$$f_i : \mathbb{R}^d \to \mathbb{R}, x \mapsto \left(\frac{2\gamma_i}{\pi} \right)^{d/4} \exp\big(-\gamma_i \|x - x_i\|^2 \big)$$

with unit covariance matrix and hyper-parameter $\gamma_i > 0$. A short calculation (see Appendix) shows that the scalar product $\langle f_i, f_j \rangle$ satisfies

$$\langle f_i, f_j \rangle = \left(\frac{2\sqrt{\gamma_i \gamma_j}}{\gamma_i + \gamma_j} \right)^{d/2} \exp\left(-\frac{\gamma_i \gamma_j}{\gamma_i + \gamma_j} \|x_i - x_j\|^2 \right).$$

If the hyper-parameter γ_i is the same ($= \gamma$) for all the Gaussians, then the resulting regularization problem coincides up to constant scaling with a Gaussian kernel support vector machine with hyper-parameter γ, i.e., to regularization in the corresponding reproducing kernel Hilbert space with the kernel function

$$k(x_i, x_j) = \exp\left(-\frac{\gamma}{2}\|x_i - x_j\|^2\right).$$

However, regularization in $L_2(\mathbb{R}^d)$ provides more freedom, namely combining Gaussians with different hyper-parameters γ_i. Note that the standard approach of linearly combining kernels with different hyper-parameters cannot provide the same variability since the weights of the different kernel functions have to be fixed in the combination, even if the weights have been optimized in a kernel learning approach as in [1].

Especially, we can simultaneously consider normalized Gaussians with different hyper-parameters centered at the same data point, e.g.,

$$f_i : \mathbb{R}^d \to \mathbb{R}, x \mapsto \left(\frac{2\gamma_i}{\pi}\right)^{d/4} \exp\big(-\gamma_i\|x - x_i\|^2\big)$$

and at the same time

$$f_i' : \mathbb{R}^d \to \mathbb{R}, x \mapsto \left(\frac{2\gamma_i'}{\pi}\right)^{d/4} \exp\big(-\gamma_i'\|x - x_i\|^2\big).$$

For example, considering two hyper-parameters γ and γ' at every data point results in twice the number of data points many basis functions.

3.2 Incremental construction

Since the number of basis functions m can also be smaller than the number of data points n we can also add the basis functions incrementally one at a time. The incremental construction can be stopped for example after a fixed number of basis functions have been added, resulting in an optimized classifier with prescribed *sparsity*, or after a prescribed classification performance (e.g., cross-validation value) has been reached, resulting in a classifier whose *sparseness* is optimized at a prescribed performance measure.

Assume we have n data points. We can start using two orthonormal basis functions e_1 and e_2, e.g., centered at two data points with opposite labels, and solve the non-linear primal SVM with all n constraints but only two variables namely the coefficients for the two basis functions. From the solution we derive a classifier $h_2 : \Omega \to \{-1, 1\}$. In a *boosting* like fashion we can pick a next data point at which we have not centered a basis function so far[1] and center the next normalized basis function e_3 (orthogonal to the previous ones) there to compute the classifier h_3. Repeating this procedure provides a sequence $e_1, \ldots, e_m$ of orthonormal basis functions and associated classifiers $h_1, \ldots, h_m$.

The incremental approach allows to use ideas of the structural risk minimization principle [13]. The basis functions span a sequence of nested hypothesis spaces $\mathcal{H}_2 \subset \mathcal{H}_3 \subset \ldots \subset \mathcal{H}_m$ of increasing complexity, where the space $\mathcal{H}_i$ is spanned by the basis functions $e_1, \ldots, e_i$. For every hypothesis space we solve a non-linear primal SVM. This provides us with optimal coefficient vectors $w_i \in \mathbb{R}^i, i = 2, \ldots, m$ and optimal $b_i \in \mathbb{R}, i = 1, \ldots, m$ from which we derive the classifiers $h_i, i = 2 \ldots, m$.

[1] There are several strategies possible to choose the next data point. One could take the data point that gets misclassified by h_2 the most, or one could take the data point that gives overall best improvement of the resulting classifier. The latter strategy is more costly of course since all data points that have not been used so far need to be tested.

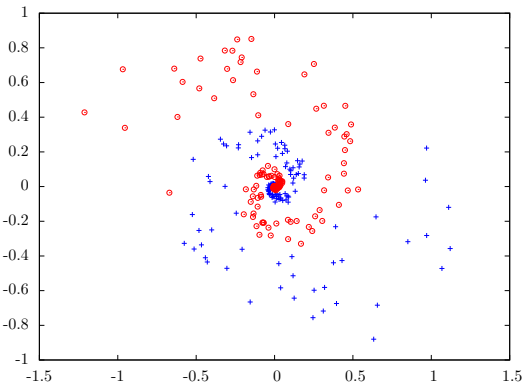

Figure 1: Synthetic data set: Two confocal spirals

Using cross-validation and sampling or a path following algorithm [8] we can even optimize the regularization parameters $c_i \geq 0, i = 2, \ldots, m$. Taking the cross validation values of the non-linear primal SVMs at the optimized regularization parameters c_i provides us with $m - 1$ values. Ideally, these values decrease with growing number of basis functions. We can monitor this progress and use it to decide when to stop adding further basis functions.

4 Experiments

In the following we report on experiments we did with basis functions (orthonormalized Gaussians) for SVMs. The main point we wanted to make in Section 3 is that the number n of data points and number m of basis functions need not to be the same. The case of more basis functions than data points is considered in experiments with "non-uniform" basis functions and the case of fewer basis functions than data points is considered in experiments on the incremental construction.

4.1 Non-uniformity

To explore the use of basis functions with different hyper-parameters we created a synthetic data set consisting of two spirals in the plane with 200 data points each. The spirals' spread increases when moving away from the center. Each point of a spiral is perturbed by Gaussian noise whose variance depends on the distance to the center. Hence, points from the two spirals close to the center are close to each other and their separation grows as they move away from the center. The data set is visualized in Figure 1.

For the standard kernelized SVM we chose a Gaussian kernel and performed a grid search for the hyper-parameter γ and the regularization parameter c using 5-fold cross validation. Figure 2 (left) shows the cross-validation value as function of the hyper-parameter γ, where the regularization parameter c has been optimized individually at every γ-value. The optimal 5-fold cross validation error of 22% misclassified points is achieved at $\gamma = 2^9$. For the non-linear primal SVM we have centered two Gaussians with hyper-parameter γ_1 and γ_2, respectively, at each data point. This time we performed a grid search over the hyper-parameter γ_1 and γ_2 and the regularization parameter c again using 5-fold cross validation. Figure 2 (right) shows the 5-fold cross validation error over a grid of values for γ_1 and γ_2. The minimal 5-fold cross validation error of 11% is obtained at $\left(\gamma_1 = 2^7, \gamma_2 = 2^{19}\right)$—

again, with optimized regularization parameter c at every pair (γ_1, γ_2). Note that due to symmetry $\gamma_1 = 2^{19}$ and $\gamma_2 = 2^7$ gives the same cross validation error.

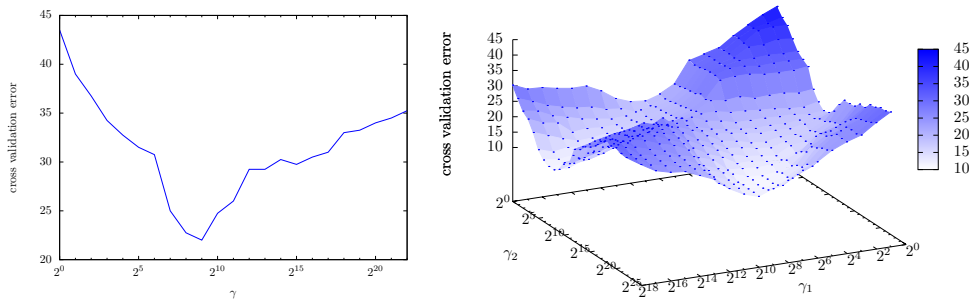

Figure 2: 5-fold cross validation error (on a logarithmic scale) for the two spirals data set (see Figure 1) using an Gaussian kernel with hyper-parameter γ (left), and two Gaussian basis functions with hyper-parameters γ_1 and γ_2, respectively, at every data point (right).

This small experiment on a toy data set indicates that using non-uniform basis functions can improve the 5-fold cross validation, here for example the cross-validation error is decreased from 22% to 11% which is a relative improvement of 50%.

We performed the same experiment on the Ionosphere data set from the UCI Machine Learning Repository [6], see also Figure 3. With a Gaussian kernel a 5-fold cross validation error of exactly 4% has been achieved for $\gamma = 2^{-2}$. Using the non-uniform basis functions as described before gives an error of approximately 3.14% at $\left(\gamma_1 = 2^{-7}, \gamma_2 = 2^{-3}\right)$, which means a relative improvement of approximately 21%.

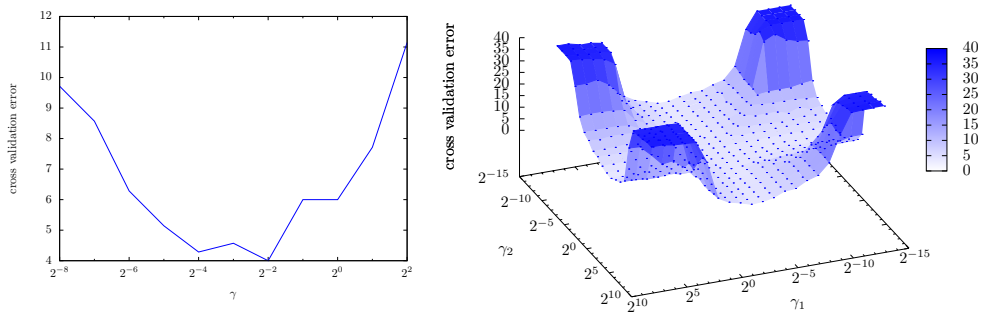

Figure 3: 5-fold cross validation error (on a logarithmic scale) for the Ionosphere data set using a Gaussian kernel with hyper-parameter γ (left) and non-uniform basis functions with hyper-parameters γ_1 and γ_2 (right).

4.2 Incremental construction

We ran experiments with the incremental construction again on the Ionosphere data set. To validate our results we compare the results with another incremental construction in the kernel case, namely by adding one data point after the other (in a well chosen manner). Note that in the kernel case the number of data points and the number of variables (basis functions) is always the same whereas these number are decoupled in the basis function case, i.e., we always use all data points but only a fixed number of basis functions for training the SVM. In the experiments we started using one basis function on a randomly chosen data point and kept adding the "most" misclassified data point using the last iteration's classifier in the kernel case, or the basis function centered at the "most" misclassified data point in the basis function case, respectively.

In the kernel case we chose a Gaussian kernel with fixed hyper-parameter $\gamma = 2^{-3}$, and in the basis function case we considered Gaussians with the same hyper-parameter $\gamma = 2^{-3}$ centered at the data points. We always performed a grid search to optimize the regularization parameter c. Figure 4 shows the 5-fold cross validation error as a function of the number of basis functions, or data points in the kernel case, respectively, comparing both approaches (always at the optimized regularization parameter c using cross-validation). As the number of basis functions or data points, respectively, increases the 5-fold cross validation error decreases. But because all data points have been used in the basis function case to train the SVM the cross validation value improves faster in this case. This indicates that it could be beneficial to chose basis functions over kernels if a classifier with given sparsity is sought for.

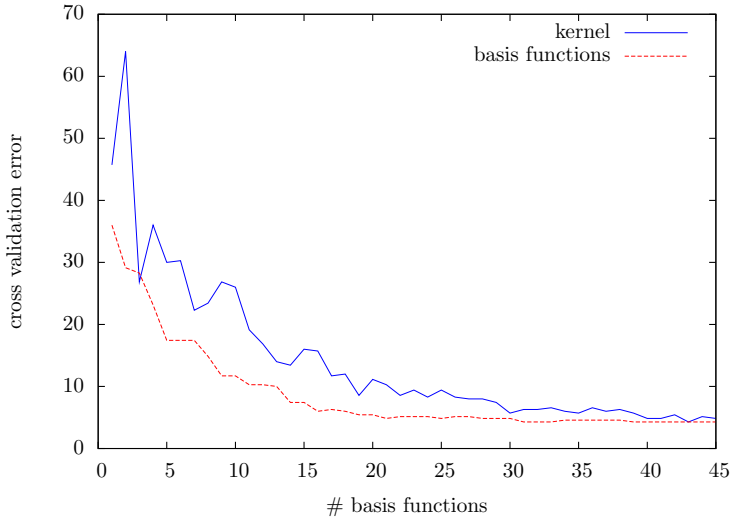

Figure 4: 5-fold cross validation error as a function of the number of basis functions / data points (kernel case) on the Ionosphere data set.

5 Discussion and conclusion

We have re-visited non-linear primal support vector machines. So far these SVMs have been mostly discussed in terms of optimization algorithms to train them. Here we have pointed out that turning non-linear in the primal offers additional flexibility

for modeling classification problems. This is mostly due to the fact that the number of basis functions does not need to be the same as the number of data points when considering basis functions instead of kernels for SVMs. We are confident that the basis function view can be benefical beyond what we have discussed here and want to explore this in future work.

The idea of using basis functions instead of kernels implicitly appears in many works, for instance in [15], where the authors model the density distribution using kernels as basis functions and provide an algorithm that runs in the primal using a logistic penalty error. The algorithm runs in iterations and at each iteration one new basis function is added in a boosting-like fashion. Each of the papers [7, 11] provides an algorithm for computing large margin classifiers that run on the primal optimization problem. Both papers noted that since their algorithms use only scalar products they can be adapted to employ non-linear kernels using the representer theorem while working in the primal setting. As discussed in Section 2 this can be achieved for any algorithm that solely works in the primal setting and hence is a property of the optimization problem cast by SVMs rather than a property of the specific algorithm. Finding an orthonormal basis by using a singular value decomposition has been described for instance in [12, 14]. But the focus of these papers is rather on speeding up kernelized machine learning algorithms by means of sparse approximation than on exploiting the orthonormal representation for modeling the problem.

State-of-the-art SVM solvers, like LIBSVM [3], empirically have a running time of only $O(n)$ on n data points to compute an optimal solution. To compute an orthonormal kernel function either through he Gram-Schmidt orthonormalization scheme or through a singular value decomposition of the kernel matrix needs $O(n^3)$ running time. However, one can solve the problem in the primal directly without orthonormalization. In this case the same running time is achieved as for standard kernelized dual solvers like LIBSVM.

Appendix

Scalar product of two Gaussians

The formulas in Section 3.1 can be all derived from the $L_2(\mathbb{R}^d)$ dot product of two (unnormalized) Gaussians,

$$\langle g_i, g_j \rangle = \int_{\mathbb{R}^d} g_i g_j \, d\mu, \tag{1}$$

with

$$g_i : \mathbb{R}^d \to \mathbb{R}, x \mapsto \exp\left(-\gamma_i \|x - x_i\|^2\right) \quad \text{and} \quad g_j : \mathbb{R}^d \to \mathbb{R}, x \mapsto \exp\left(-\gamma_j \|x - x_j\|^2\right).$$

To evaluate the integral in Equation (1) we consider the following product

$$\begin{aligned} g_i(x) g_j(x) &= \exp\left(-(\gamma_i + \gamma_j)\|x\|^2 + 2x^T(\gamma_i x_i + \gamma_j x_j) - \gamma_i \|x_i\|^2 - \gamma_j \|x_j\|^2\right) \\ &= \exp\left(-(\gamma_i + \gamma_j)\left\|x - \left(\frac{\gamma_i x_i}{\gamma_i + \gamma_j} + \frac{\gamma_j x_j}{\gamma_i + \gamma_j}\right)\right\|^2 - \frac{\gamma_i \gamma_j}{\gamma_i + \gamma_j}\|x_i - x_j\|^2\right) \\ &= \exp\left(-(\gamma_i + \gamma_j)\left\|x - \left(\frac{\gamma_i x_i}{\gamma_i + \gamma_j} + \frac{\gamma_j x_j}{\gamma_i + \gamma_j}\right)\right\|^2\right) \exp\left(-\frac{\gamma_i \gamma_j}{\gamma_i + \gamma_j}\|x_i - x_j\|^2\right). \end{aligned}$$

Using

$$\int_{\mathbb{R}^d} \exp\left(-(\gamma_i + \gamma_j)\left\|x - \left(\frac{\gamma_i x_i}{\gamma_i + \gamma_j} + \frac{\gamma_j x_j}{\gamma_i + \gamma_j}\right)\right\|^2\right) d\mu = \left(\frac{\pi}{\gamma_i + \gamma_j}\right)^{d/2}$$

we obtain

$$\langle g_i, g_j \rangle = \left(\frac{\pi}{\gamma_i + \gamma_j} \right)^{d/2} \exp\left(-\frac{\gamma_i \gamma_j}{\gamma_i + \gamma_j} \|x_i - x_j\|^2 \right).$$

References

[1] Francis R. Bach, Gert R. G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2004.

[2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory (COLT)*, pages 144–152, 1992.

[3] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[4] Olivier Chapelle. Training a Support Vector Machine in the Primal. *Neural Computation*, 19(5):1155–1178, 2007.

[5] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[6] A. Frank and A. Asuncion. UCI Machine Learning Repository, 2010.

[7] Yoav Freund and Robert E. Schapire. Large Margin Classification Using the Perceptron Algorithm. In *Machine Learning*, pages 277–296, 1998.

[8] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415, 2004.

[9] S. Sathiya Keerthi and Dennis DeCoste. A Modified Finite Newton Method for Fast Solution of Large Scale Linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

[10] George S. Kimeldorf and Grace Wahba. A Correspondence Between Bayesian Estimation on Stochastic Processes and Smoothing by Splines. *The Annals of Mathematical Statistics*, 41:595–502, 1970.

[11] Yoram Singer and Nathan Srebro. PEGASOS: primal estimated sub-gradient solver for SVM. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 807–814, 2007.

[12] Alex J. Smola and Bernhard Schölkopf. Sparse Greedy Matrix Approximation for Machine Learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 911–918, 2000.

[13] Vladimir N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[14] Christopher K. I. Williams and Matthias Seeger. Using the Nyström Method to Speed Up Kernel Machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 682–688, 2000.

[15] Ji Zhu and Trevor Hastie. Kernel Logistic Regression and the Import Vector Machine. In *Journal of Computational and Graphical Statistics*, pages 1081–1088. MIT Press, 2001.