Available online at www.sciencedirect.com

SCIENCE DIRECT®

Neurocomputing 69 (2006) 730–742

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Support vector machine for functional data classification

Fabrice Rossi[a,*], Nathalie Villa[b]

[a]*Projet AxIS, INRIA-Rocquencourt, Domaine de Voluceau, Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France*
[b]*Equipe GRIMM, Université Toulouse Le Mirail, 5 allées A. Machado, 31058 Toulouse Cedex 1, France*

Available online 19 January 2006

Abstract

In many applications, input data are sampled functions taking their values in infinite-dimensional spaces rather than standard vectors. This fact has complex consequences on data analysis algorithms that motivate their modifications. In fact most of the traditional data analysis tools for regression, classification and clustering have been adapted to functional inputs under the general name of functional data analysis (FDA). In this paper, we investigate the use of support vector machines (SVMs) for FDA and we focus on the problem of curve discrimination. SVMs are large margin classifier tools based on implicit nonlinear mappings of the considered data into high-dimensional spaces thanks to kernels. We show how to define simple kernels that take into account the functional nature of the data and lead to consistent classification. Experiments conducted on real world data emphasize the benefit of taking into account some functional aspects of the problems.

Keywords: Functional data analysis; Support vector machine; Classification; Consistency

1. Introduction

In many real world applications, data should be considered as discretized functions rather than as standard vectors. In these applications, each observation corresponds to a mapping between some conditions (that might be implicit) and the observed response. A well studied example of those functional data is given by spectrometric data (see Section 6.3): each spectrum is a function that maps the wavelengths of the illuminating light to the corresponding absorbances (the responses) of the studied sample. Other natural examples can be found in voice recognition area (see Sections 6.1 and 6.2) or in meteorological problems, and more generally, in multiple time series analysis where each observation is a complete time series.

The direct use of classical models for this type of data faces several difficulties: as the inputs are discretized functions, they are generally represented by high-dimensional vectors whose coordinates are highly correlated. As a consequence, classical methods lead to ill-posed problems, both on a theoretical point of view (when working in functional spaces that have infinite dimension) and on a practical one (when working with the discretized functions). The goal of functional data analysis (FDA) is to use, in data analysis algorithms, the underlying functional nature of the data: many data analysis methods have been adapted to functions (see [29] for a comprehensive introduction to functional data analysis and a review of linear methods). While the original papers on FDA focused on linear methods such as principal component analysis [10,8,9,2] and the linear model [30,16,18], nonlinear models have been studied extensively in recent years. This is the case, for instance, of most neural network models [13,31–33].

In the present paper, we adapt support vector machines (SVMs, see e.g. [42,7]) to functional data classification (the paper extends results from [34,44]). We show in particular both the practical and theoretical advantages of using functional kernels, which are kernels that take into account the functional nature of the data. On a practical point of view, those kernels allow us to take advantage of the expert knowledge on the data. From a theoretical point of view, a specific type of functional kernel allows the construction of a consistent training procedure for functional SVMs.

*Corresponding author. Tel.: +33 1 39 63 54 45; fax: +33 1 39 63 58 92.
E-mail addresses: fabrice.rossi@inria.fr (F. Rossi), villa@univ-tlse2.fr (N. Villa).

0925-2312/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2005.12.010

The paper is organized as follows: Section 2 presents the basic idea of functional data classification and explains why it generally leads to ill-posed problems. Section 3 provides a short introduction to SVMs and explains why their generalization to FDA can lead to particular problems. Section 4 describes several functional kernels and explains how they can be practically computed while Section 5 presents a consistency result for some of them. Finally, Section 6 illustrates the various approaches presented in the paper on real data sets.

2. Functional data analysis

2.1. Functional data

To simplify the presentation, this article focuses on functional data for which each observation is described by one function from $\mathbb{R}$ to $\mathbb{R}$. Extension to the case of several real valued functions is straightforward. More formally, if μ denotes a known finite positive Borel measure on $\mathbb{R}$, an observation is an element of $L^2(\mu)$, the Hilbert space of μ-square-integrable real valued functions defined on $\mathbb{R}$. In some situations, additional regularity assumptions (e.g., existence of derivatives) will be needed.

However, almost all the developments of this paper are not specific to functions and use only the Hilbert space structure of $L^2(\mu)$. We will therefore denote as $\mathcal{X}$, an arbitrary Hilbert space and $\langle \cdot, \cdot \rangle$ the corresponding inner product. Additional assumptions on $\mathcal{X}$ will be given on a case by case basis. As stated above, the most common situation will of course be $\mathcal{X} = L^2(\mu)$ with $\langle u, v\rangle = \int uv\,\mathrm{d}\mu$.

2.2. Data analysis methods for Hilbert spaces

First it should be noted that many data analysis algorithms can be written so as to apply, at least from a theoretical point of view, to arbitrary Hilbert spaces. This is obviously the case, for instance, for distance-based algorithms such as the k-nearest neighbor method. Indeed, this algorithm uses only the fact that distances between observations can be calculated. Obviously, it can be applied to Hilbert spaces using the distance induced by the inner product. This is also the case for methods directly based on inner products such as multi-layer perceptrons (see [35,36,41] for a presentation of multi-layer perceptrons with almost arbitrary input spaces, including Hilbert spaces).

However, functional spaces have infinite dimension and a basic transposition of standard algorithms introduces both theoretical and practical difficulties. In fact, some simple problems in $\mathbb{R}^d$ become ill-posed in $\mathcal{X}$ when the space has infinite dimension, even from a theoretical point of view.

Let us consider for instance the linear regression model in which a real valued target variable Y is modeled by $E(Y|X) = H(X)$ where H is a linear continuous operator defined on the input space. When X has values in $\mathbb{R}^d$ (i.e., $\mathcal{X} = \mathbb{R}^d$), H can be easily estimated by the least squares method that leads to the inversion of the covariance matrix of X. In practice, problems might appear when d is not small compared to N, the number of available examples, and regularization techniques should be used (e.g., ridge regression [21]). When X has values in a Hilbert space, the problem is ill-posed because the covariance of X is a Hilbert–Schmidt operator and thus has no continuous inverse; direct approximation of the inverse of this operator is then problematic as it does not provide a consistent estimate (see [4]).

To overcome the infinite-dimensional problem, most of FDA methods so far have been constructed thanks to two general principles: *filtering* and *regularization*. In the filtering approach, the idea is to use representation methods that allow to work in finite dimensions (see for instance [4] for the functional linear model and [3] for a functional k-nearest neighbor method). In the regularization approach, the complexity of the solution is constrained thanks to smoothness constraints. For instance, building a linear model in a Hilbert space consists in finding a function $h \in L^2(\mu)$ such that $E(Y|X) = \langle h, X\rangle$. In the regularization approach, h is chosen among smooth candidates (for instance twice derivable functions with minimal curvature), see e.g. [18,26,5]. Other examples of the regularization approach include smooth principal component analysis [27] and penalized canonical component analysis [23]. A comparison of filtering and regularization approaches for a semi-parametric model used in curve discrimination can be found in [14].

Using both approaches, a lot of data analysis algorithms have been successfully adapted to functional data. Our goal in the present paper is to study the case of SVMs, mainly thanks to a filtering approach.

3. Support vector machines for FDA

3.1. Support vector machines

We give, in this section, a very brief presentation of SVMs that is needed for the definition of their functional versions. We refer the reader to e.g. [7] for a more comprehensive presentation. As stated in Section 2.1, $\mathcal{X}$ denotes an arbitrary Hilbert space. Our presentation of SVM departs from the standard introduction because it assumes that the observations belong to $\mathcal{X}$ rather than to a $\mathbb{R}^d$. This will make clear that the definition of SVM on arbitrary Hilbert spaces is not the difficult part in the construction of functional SVM. We will discuss problems related to the functional nature of the data in Section 3.2.

Our goal is to classify data into two predefined classes. We assume given a learning set, i.e. N examples $(x_1, y_1), \ldots, (x_N, y_N)$ which are i.i.d. realizations of the random variable pair (X, Y) where X has values in $\mathcal{X}$ and Y in $\{-1, 1\}$, i.e. Y is the class label for X which is the observation.

3.1.1. Hard margin SVM

The principle of SVM is to perform an affine discrimination of the observations with maximal margin, that is to find an element $w \in \mathcal{X}$ with a minimum norm and a real value b, such that $y_i(\langle w, x_i\rangle + b) \geqslant 1$ for all i. To do so, we have to solve the following quadratic programming problem:

$$\begin{aligned} &\min_{w,b} \quad \langle w, w\rangle \\ &\text{s.t.} \quad y_i(\langle w, x_i\rangle + b) \geqslant 1, \quad 1 \leqslant i \leqslant N. \end{aligned} \qquad (P_0)$$

The classification rule associated with (w, b) is simply $f(x) = \text{sign}(\langle w, x\rangle + b)$. In this situation (called the hard margin SVM), we require the rule to have zero error on the learning set.

3.1.2. Soft margin SVM

In practice, the solution provided by problem (P_0) is not very satisfactory. Firstly, perfectly linearly separable problems are quite rare, partly because nonlinear problems are common, but also because noise can turn a linearly separable problem into a nonseparable one. Secondly, choosing a classifier with maximal margin does not prevent overfitting, especially in very high-dimensional spaces (see e.g. [19] for a discussion about this point).

A first step in solving this problem is to allow some classification errors on the learning set. This is done by replacing (P_0) by its soft margin version, i.e., by the problem:

$$\begin{aligned} &\min_{w,b,\xi} \quad \langle w, w\rangle + C\sum_{i=1}^{N}\xi_i \\ &\text{s.t.} \quad y_i(\langle w, x_i\rangle + b) \geqslant 1 - \xi_i, \ 1 \leqslant i \leqslant N, \\ &\qquad\ \ \xi_i \geqslant 0, \ 1 \leqslant i \leqslant N. \end{aligned} \qquad (P_C)$$

Classification errors are allowed thanks to the slack variables ξ_i. The C parameter acts as an inverse regularization parameter. When C is small, the cost of violating the hard margin constraints, i.e., the cost of having some $\xi_i > 0$ is small and a solution with a small norm for w will be chosen, even if that leads to many classification errors. On the contrary, when C is large, classification errors dominate and (P_C) gets closer to (P_0).

3.1.3. Nonlinear SVM

As noted in the previous section, some classification problems do not have a satisfactory linear solution but have a nonlinear one. Nonlinear SVMs are obtained by transforming the original data. Assume given an Hilbert space $\mathcal{H}$ (and denote $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ the corresponding inner product) and a function ϕ from $\mathcal{X}$ to $\mathcal{H}$ (this function is called a *feature map*). A linear SVM in $\mathcal{H}$ can be constructed on the data set $(\phi(x_1), y_1), \ldots, (\phi(x_N), y_N)$. If ϕ is a nonlinear mapping, the classification rule $f(x) = \text{sign}(\langle w, \phi(x)\rangle_{\mathcal{H}} + b)$ is also nonlinear.

In order to obtain the linear SVM in $\mathcal{H}$ one has to solve the following optimization problem:

$$\begin{aligned} &\min_{w,b,\xi} \quad \langle w, w\rangle_{\mathcal{H}} + C\sum_{i=1}^{N}\xi_i \\ &\text{s.t.} \quad y_i(\langle w, \phi(x_i)\rangle_{\mathcal{H}} + b) \geqslant 1 - \xi_i, \ 1 \leqslant i \leqslant N, \\ &\qquad\ \ \xi_i \geqslant 0, \ 1 \leqslant i \leqslant N. \end{aligned} \qquad (P_{C,\mathcal{H}})$$

It should be noted that this feature mapping allows one to define SVM on almost arbitrary input spaces.

3.1.4. Dual formulation and Kernels

Solving problems (P_C) or $(P_{C,\mathcal{H}})$ might seem very difficult at first, because $\mathcal{X}$ and $\mathcal{H}$ are arbitrary Hilbert spaces and can therefore have very high or even infinite dimension (when $\mathcal{X}$ is a functional space for instance). However, each problem has a dual formulation. More precisely, (P_C) is equivalent to the following optimization problem (see [24]):

$$\begin{aligned} &\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle \\ &\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \\ &\qquad\ \ 0 \leqslant \alpha_i \leqslant C, \quad 1 \leqslant i \leqslant N. \end{aligned} \qquad (D_C)$$

This result applies to the original problem in which data are not mapped into $\mathcal{H}$, but also to the mapped data, i.e., $(P_{C,\mathcal{H}})$ is equivalent to a problem $(D_{C,\mathcal{H}})$ in which the x_i are replaced by $\phi(x_i)$ and in which the inner product of $\mathcal{H}$ is used. This leads to

$$\begin{aligned} &\max_{\alpha} \quad \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \langle \phi(x_i), \phi(x_j)\rangle_{\mathcal{H}} \\ &\text{s.t.} \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \\ &\qquad\ \ 0 \leqslant \alpha_i \leqslant C, \ 1 \leqslant i \leqslant N. \end{aligned} \qquad (D_{C,\mathcal{H}})$$

Solving $(D_{C,\mathcal{H}})$ rather than $(P_{C,\mathcal{H}})$ has two advantages. The first positive aspect is that $(D_{C,\mathcal{H}})$ is an optimization problem in $\mathbb{R}^N$ rather than in $\mathcal{H}$ which can have infinite dimension (the same is true for $\mathcal{X}$).

The second important point is linked to the fact that the optimal classification rule can be written $f(x) = \text{sign}(\sum_{i=1}^{N}\alpha_i y_i \langle\phi(x_i), \phi(x)\rangle_{\mathcal{H}} + b)$. This means that both the optimization problem and the classification rule do not make direct use of the transformed data, i.e. of the $\phi(x_i)$. All the calculations are done through the inner product in $\mathcal{H}$, more precisely through the values $\langle\phi(x_i), \phi(x_j)\rangle_{\mathcal{H}}$. Therefore, rather than choosing directly $\mathcal{H}$ and ϕ, one can provide a so-called *Kernel function K* such that $K(x_i, x_j) = \langle\phi(x_i), \phi(x_j)\rangle_{\mathcal{H}}$ for a given pair $(\mathcal{H}, \phi)$.

In order that K corresponds to an actual inner product in a Hilbert space, it has to fulfill some conditions. K has to be symmetric and positive definite, that is, for every N, $x_1, \ldots, x_N$ in $\mathcal{X}$ and $\alpha_1, \ldots, \alpha_N$ in $\mathbb{R}$,

$\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j K(x_i,x_j)\geqslant 0$. If K satisfies those conditions, according to Moore–Aronszajn theorem [1], there exists a Hilbert space $\mathscr{H}$ and feature map ϕ such that $K(x_i,x_j)=\langle\phi(x_i),\phi(x_j)\rangle_{\mathscr{H}}$.

3.2. *The case of functional data*

The short introduction to SVM proposed in the previous section has clearly shown that defining linear SVM for data in a functional space is as easy as for data in $\mathbb{R}^d$, because we only assume that the input space is a Hilbert space. By the dual formulation of the optimization problem (P_C), a software implementation of linear SVM on functional data is even possible, by relying on numerical quadrature methods to calculate the requested integrals (inner product in $L^2(\mu)$, cf. Section 4.3).

However, the functional nature of the data has some effects. It should be first noted that in infinite-dimensional Hilbert spaces, the hard margin problem (P_0) has always a solution when the input data are in general position, i.e., when N observations span a N dimensional subspace of $\mathscr{X}$. A very naive solution would therefore consist of avoiding soft margins and nonlinear kernels. This would not give very interesting results in practice because of the lack of regularization (see [19] for some examples in very high dimension spaces, as well as Section 6.1).

Moreover, the linear SVM with soft margin can also lead to poor performance. It is indeed well known (see e.g. [20]) that problem (P_C) is equivalent to the following unconstrained optimization problem:

$$\min_{w,b}\frac{1}{N}\sum_{i=1}^{N}\max(0,1-y_i(\langle w,x_i\rangle+b))+\lambda\langle w,w\rangle, \qquad (R_\lambda)$$

with $\lambda=1/CN$. This way of viewing (P_C) emphasizes the regularization aspect (see also [37,38,12]) and links the SVM model to ridge regression [21]. As shown in [17], the penalization used in ridge regression behaves poorly with functional data. Of course, the loss function used in SVM (the *hinge loss*, i.e., $h(u,v)=\max(0,1-uv)$) is different from the quadratic loss used in ridge regression and therefore no firm conclusion can be drawn from experiments reported in [17]. However they show that we might expect bad performances with the linear SVMs applied directly to functional data. We will see in Sections 6.1 and 6.2 that the efficiency of ridge regularization seems to be linked with the number of features of the data: it does not behave very well when the number of discretization points is very large and thus leads us to approximate the ridge penalty by a dot product in a very high-dimensional space (see also Section 4.3).

It is therefore interesting to consider nonlinear SVM for functional data, by introducing adapted kernels. As pointed out in e.g. [12], $(P_{C,\mathscr{H}})$ is equivalent to

$$\min_{f\in\mathscr{H}}\frac{1}{N}\sum_{i=1}^{N}\max(0,1-y_if(x_i))+\lambda\langle f,f\rangle_{\mathscr{H}}. \qquad (R_{\lambda,\mathscr{H}})$$

Using a kernel therefore corresponds both to replacing a linear classifier by a nonlinear one, but also to replacing the ridge penalization by a penalization induced by the kernel which might be more adapted to the problem (see [38] for links between regularization operators and kernels). The applications presented in Section 6 illustrate this fact.

4. Kernels for FDA

4.1. *Classical kernels*

Many standard kernels for $\mathbb{R}^d$ data are based on the Hilbert structure of $\mathbb{R}^d$ and can therefore be applied to any Hilbert space. This is the case for instance of the Gaussian kernel (based on the norm in $\mathscr{X}: K(u,v)=\mathrm{e}^{-\sigma\|u-v\|^2}$) and of the polynomial kernels (based on the inner product in $\mathscr{X}: K(u,v)=(1+\langle u,v\rangle)^D$). Obviously, the only practical difficulty consists in implementing the calculations needed in $\mathscr{X}$ so as to evaluate the chosen kernel (the problem also appears for the plain linear "kernel", i.e. when no feature mapping is done). Section 4.3 discusses this point.

4.2. *Using the functional nature of the data*

While the functional versions of standard kernels can provide an interesting library of kernels, they do not take advantage of the functional nature of the data (they use only the Hilbert structure of $L^2(\mu)$). Kernels that use the fact that we are dealing with functions, rather than vectorial data, are nevertheless quite easy to define.

A standard method consists of introducing kernels that are made by a composition of a simple feature map with a standard kernel. More formally, we use a transformation operator P from $\mathscr{X}$ to another space $\mathscr{D}$ on which a kernel K is defined. The actual kernel Q on $\mathscr{X}$ is defined as $Q(u,v)=K(P(u),P(v))$ (if K is a kernel, then so is Q).

4.2.1. *Functional transformations*

In some application domains, such as chemometrics, it is well known that the shape of a spectrum (which is a function) is sometimes more important than its actual mean value. Several transformations can be proposed to deal with this kind of data. For instance, if μ is a finite measure (i.e., $\mu(\mathbb{R})<\infty$), a centering transformation can be defined as the following mapping from $L^2(\mu)$ to itself:

$$C(u)=u-\frac{1}{\mu(\mathbb{R})}\int u\,\mathrm{d}\mu.$$

A normalization mapping can also be defined:

$$N(u)=\frac{1}{\|C(u)\|}C(u).$$

If the functions are smooth enough, i.e., if we restrict ourselves to a Sobolev space $W^{s,2}$, then some derivative transformations can be used: the Sobolev space $W^{s,2}$, also denoted H^s, is the Hilbert space of functions which have L^2

derivatives up to the order s (in the sense of distribution theory). For instance, with $s \geqslant 2$, we can use the second derivative to focus on the curvature of the functions; this is particularly useful in near infrared spectrometry (see e.g., [31,33], and Section 6.3).

4.2.2. Projections

Another type of transformation can be used in order to define adapted kernels. The idea is to reduce the dimensionality of the input space, that is to apply the standard filtering approach of FDA. We assume given a d-dimensional subspace V_d of $\mathscr{X}$ and an orthonormal basis of this space denoted $\{\Psi_j\}_{j=1,\dots,d}$. We define the transformation P_{V_d} as the orthogonal projection on V_d,

$$P_{V_d}(x) = \sum_{j=1}^{d} \langle x, \Psi_j \rangle \Psi_j.$$

$(V_d, \langle\cdot,\cdot\rangle_{\mathscr{X}})$ is isomorphic to $(\mathbb{R}^d, \langle\cdot,\cdot\rangle_{\mathbb{R}^d})$ and therefore one can use a standard $\mathbb{R}^d$ SVM on the vector data $(\langle x, \Psi_1\rangle, \dots, \langle x, \Psi_d\rangle)$. This means that K can be any kernel adapted to vector data. In the case where K is the usual dot product in $\mathbb{R}^d$, this kernel is known as the empirical kernel map (see [43] for further details in the field of protein analysis).

Obviously, this approach is not restricted to functional data, but the choice of V_d can be directed by expert knowledge on the considered functions and such that it takes advantage of the functional nature of the data. We outline here two possible solutions, the first based on an orthogonal basis and the second on a B-spline basis.

If $\mathscr{X}$ is separable, it has a Hilbert basis, i.e., a complete orthonormal system $\{\Psi_j\}_{j\geqslant 1}$. Therefore, one can define V_d as the space spanned by $\{\Psi_j\}_{j=1,\dots,d}$. The choice of the basis can be based on expert considerations. Good candidates include the Fourier basis and the wavelet basis. If the signal is known to be nonstationary, a wavelet based representation might for instance give better results than a Fourier representation. Once the basis is chosen, an optimal value for d can be derived from the data, as explained in Section 5, in such a way that the obtained SVM has some consistency properties. Moreover, this projection approach gives good results in practice (see Section 6.1).

Another solution is to choose a projection space that has interesting practical properties, for instance a spline space with its associated B-spline bases. The smoothness of spline functions can be chosen a priori so as to enforce expert knowledge on the functions. For instance, near infrared spectra are smooth because of the physical properties of the light transmission (and reflection). By using a spline representation of the spectra, we replace original unconstrained observations by C^k approximations (k depends on what kind of smoothness hypothesis are reasonable). This projection can also be combined with a derivative transformation operation (as proposed in Section 4.2.1).

4.3. Functional data in practice

In practice, the functions $(x_i)_{1\leqslant i\leqslant N}$ are never perfectly known. It is therefore difficult to implement exactly the functional kernels described in this section. The best situation is that in which d discretization points have been chosen in $\mathbb{R}$, $(t_k)_{1\leqslant k\leqslant d}$, and each function x_i is described by a vector of $\mathbb{R}^d$, $(x_i(t_1), \dots, x_i(t_d))$. In this situation, a simple solution consists in assuming that standard operations in $\mathbb{R}^d$ (linear combinations, inner product and norm) are good approximations of their counterparts in the considered functional space. When the sampling is regular, this is equivalent to applying standard SVMs to the vector representation of the functions (see Section 6 for real world examples of this situation). When the sampling is not regular, integrals should be approximated via a quadrature method that will take into account the relative positions of the sampling points.

In some application domains, especially medical ones (e.g. [22]), the situation is not as good. Each function is in general badly sampled: the number and the location of discretization points depend on the function and therefore a simple vector model is no longer feasible. A possible solution in this context consists of constructing an approximation of x_i based on its observation values (thanks to e.g., B-splines) and then to work with the reconstructed functions (see [29,33] for details).

The function approximation tool used should be simple enough to allow easy implementation of the required operations. This is the case for instance for B-splines that also allow derivative calculations and an easy implementation of the kernels described in Section 4.2.1. It should be noted that spline approximation is different from projection on a spline subspace. Indeed each sampled function could be approximated on a different B-spline basis, whereas the projection operator proposed in Section 4.2.2 requires an unique projection space and therefore the same B-spline basis for each input function. In other words, the spline approximation is a convenient way of representing functions (see Section 6.3 for an application to real world data), whereas the spline projection corresponds to a data reduction technique. Both aspects can be combined.

5. Consistency of functional SVM

5.1. Introduction

In this section we study one of the functional kernels described above and show that it can be used to define a consistent classifier for functional data. We first introduce some notations and definitions.

Our goal is to define a training procedure for functional SVM such that the asymptotic generalization performances of the constructed model is optimal. We define as usual the generalization error of a classifier f by the probability of misclassification:

$$L(f) = \mathbb{P}(f(X) \neq Y).$$

The minimal generalization error is the Bayes error achieved by the optimal classifier f^* given by

$$f^*(x) = \begin{cases} 1 & \text{when } \mathbb{P}(Y=1|X=x)>1/2, \\ -1 & \text{otherwise.} \end{cases}$$

We denote $L^* = L(f^*)$ the optimal Bayes error. Of course, the closer the error of a classifier is from L^*, the better its generalization ability is.

Suppose that we are given a learning sample of size N defined as in Section 3.1. A learning procedure is an algorithm which allows the construction, from this learning sample, of a classification rule f_N chosen from a set of admissible classifiers. This algorithm is said to be consistent if

$$L(f_N) \overset{N\to+\infty}{\longrightarrow} L^*.$$

It should be noted that when the data belong to $\mathbb{R}^d$, SVMs do not always provide consistent classifiers. Some sufficient conditions have been given in [40]: the input data must belong to a compact subset of $\mathbb{R}^d$, the regularization parameter (C in $(P_{C,\mathscr{H}})$) has to be chosen in specific way (in relation to N and to the type of kernel used) and the kernel must be *universal* [39]. If ϕ is the feature map associated with a kernel K, the kernel is universal if the set of all functions of the form $x\mapsto\langle w,\phi(x)\rangle$ for $w\in\mathscr{H}$ is dense in the set of all continuous functions defined on the considered compact subset. In particular, the Gaussian kernel with any $\sigma>0$ is universal for all compact subsets of $\mathbb{R}^d$ (see [40] for further details and the proof of Theorem 1 for the precise statement on C).

5.2. A learning algorithm for functional SVM

The general methodology proposed in [3] allows us to turn (with some adaptations) a consistent algorithm for data in $\mathbb{R}^d$ into a consistent algorithm for data in $\mathscr{X}$, a separable Hilbert space. We describe in this section the adapted algorithm based on the SVM.

The methodology proposed in [3] is based on projection operators described in Section 4.2.2, more precisely on the usage of a Hilbert basis of $\mathscr{X}$. In order to build a SVM classifier based on N examples, one need to choose from the data several parameters (in addition to the weights $\{\alpha_i\}_{1\leqslant i\leqslant N}$ and b in problem $(D_{C,\mathscr{H}})$):

(1) the projection size parameter d, i.e., the dimension of the subset V_d on which the functions are projected before being submitted to the SVM (recall that V_d is the space spanned by $\{\Psi_j\}_{j=1,\dots,d}$);
(2) C, the regularization parameter;
(3) the fully specified kernel K, that is the type of the universal kernel (Gaussian, exponential, etc.) but also any parameters of this kernel such as σ for the Gaussian kernel $K(u,v)=\mathrm{e}^{-\sigma^2\|u-v\|^2}$.

Let us denote $\mathscr{A}$ a set of lists of parameters to explore (those parameters are sometimes called hyper-parameters to emphasize on the fact that they are optimized differently from the weights on the SVM). Section 5.3 gives some practical examples for $\mathscr{A}$. It should be noted that $\mathscr{A}$ is not the full space of hyper-parameters, but only a set of hyper-parameters.

Following [3] we use a validation approach to choose the best list of parameters $a\in\mathscr{A}$ and in fact the best classifier on the validation set. The data are split into two sets: a training set $\{(x_i,y_i), i=1,\dots,l_N\}$ and a validation set $\{(x_i,y_i), i=l_N+1,\dots,N\}$. For each fixed list a of parameters, the training set $\{(x_i,y_i), i=1,\dots,l_N\}$ is used to calculate the SVM classification rule $f_a(x)=\operatorname{sign}(\sum_{i=1}^{l_N}\alpha_i^* y_i K(P_{V_d}(x),P_{V_d}(x_i))+b^*)$ where $(\{\alpha_i^*\}_{1\leqslant i\leqslant l_N}, b^*)$ is the solution of $(D_{C,\mathscr{H}})$ applied to the projected data $\{P_{V_d}(x_i), i=1,\dots,l_N\}$ (please note that formally everything should be indexed by a, for instance one should write K_a rather than K, but the indexes have been omitted here for the sake of notational convenience).

The validation set is used to select the optimal value of a in $\mathscr{A}$, a^*, according to estimation of the generalization error based on a penalized empirical error, that is, we define

$$a^* = \arg\min_{a\in\mathscr{A}} \widehat{L}(f_a) + \frac{\lambda_a}{\sqrt{N-l_N}},$$

where

$$\widehat{L}(f_a) = \frac{1}{N-l_N}\sum_{n=l_N+1}^{N} \mathbb{I}_{\{f_a(x_n)\neq y_n\}},$$

and λ_a is a penalty term used to avoid the selection of the most complex models (i.e., the one with the highest d in general). The classifier f_N is then chosen as $f_N=f_{a^*}$.

5.3. Consistency

Under some conditions on $\mathscr{A}$, the algorithm proposed in the previous section is consistent. We assume given a fixed Hilbert basis of the separable Hilbert space $\mathscr{X}$, $\{\Psi_j\}_{j\geqslant 1}$. When the dimension of the projection space V_d is chosen, a fully specified kernel K has to be chosen from a finite set of kernels, $\mathscr{J}_d$. The regularization parameter C can be chosen from a bounded interval of the form $[0,\mathscr{C}_d]$, for instance thanks to the algorithm proposed in [19] that allows to calculate the validation performances for all values of C in a finite time. Therefore, the set $\mathscr{A}$ can be written $\bigcup_{d\geqslant 1}\{d\}\times\mathscr{J}_d\times[0,\mathscr{C}_d]$. An element of $\mathscr{A}$ is a triple $a=(d,K,C)$ that specifies the projection operator P_{V_d}, the kernel K (including all its parameters) and the regularization constant C.

Let us first define, for all $\varepsilon>0$, $\mathscr{N}(\mathscr{H},\varepsilon)$ the covering number of the Hilbert space $\mathscr{H}$ which is the minimum number of balls with radius ε that are needed to cover the whole space $\mathscr{H}$ (see e.g., [11, Chapter 28]). Note that for the SVM, as $\mathscr{H}$ is induced by a kernel K, this number is closely related to the kernel (in particular because the norm used to define the balls is induced by the inner product of

$\mathcal{H}$, that is by K itself); in this case, we will then denote the covering number $\mathcal{N}(K, \varepsilon)$. For example, Gaussian kernels are known to induce feature spaces with covering number of the form $\mathcal{O}(\varepsilon^{-d})$ where d is the dimension of the input space (see [40]).

Then we have:

Theorem 1. *We assume that X takes its values in a bounded subspace of the separable Hilbert space $\mathcal{X}$. We suppose that,*

$$\forall d \geqslant 1, \mathcal{J}_d \text{ is a finite set},$$
$$\exists K_d \in \mathcal{J}_d \text{ such that}: \ K_d \text{ is universal},$$
$$\exists \nu_d > 0: \ \mathcal{N}(K_d, \varepsilon) = \mathcal{O}(\varepsilon^{-\nu_d}),$$
$$\mathcal{C}_d > 1,$$

and that

$$\sum_{d \geqslant 1} |\mathcal{J}_d| \mathrm{e}^{-2\lambda_d^2} < +\infty,$$

and finally that

$$\lim_{N \to +\infty} l_N = +\infty, \quad \lim_{N \to +\infty} N - l_N = +\infty,$$

$$\lim_{N \to +\infty} \frac{l_N \log(N - l_N)}{N - l_N} = 0.$$

Then, the functional SVM $f_N = f_{a^*}$ chosen as described in Section 5.2 (where a^* is optimal in $\mathcal{A} = \bigcup_{d \geqslant 1} \{d\} \times \mathcal{J}_d \times [0, \mathcal{C}_d]$) is consistent that is

$$L(f_N) \overset{N \to +\infty}{\longrightarrow} L^*.$$

The proof of this result is given in Appendix A. It is close to the proof given in [3] except that in [3] the proof follows from an oracle inequality given for a finite grid search model. The grid search is adapted to the classifier used in [3] (a k-nearest neighbor method), but not to our setting. Our result includes the search for a parameter C which can belong to an infinite and noncountable set; this can be done by the use of the shatter coefficient of a particular class of linear classifiers which provides the behavior of the classification rule on a set of $N - l_N$ observations (see [11]).

As pointed out before, the Gaussian kernel satisfies the hypothesis of the theorem. Therefore, if $\mathcal{J}_d$ contains a Gaussian kernel for all d, then consistency of the whole procedure is guaranteed. Other nonuniversal kernels can of course be included in the search for the optimal model.

Remark 1. Note that, in this theorem, the sets $\mathcal{J}_d$ and $[0, \mathcal{C}_d]$ depend on d: this does not influence the consistency of the method. In fact, one could have chosen the same set for every d, and $\mathcal{J}_d$ could also contain a single Gaussian kernel with any parameter $\sigma > 0$. In practice however, this additional flexibility is very useful to adapt the model to the data, for instance by choosing an optimal value for σ with a Gaussian kernel from the validation set.

6. Applications

We present, in this section, several applications of the functional SVM models described above to real world data. The first two applications illustrate the consistent methodology introduced in Section 5.2: one has an input variable with a high number of discretization points and the second has far fewer discretization points. Those applications show that more benefits are obtained from the functional approach when the data can be reasonably considered as functions, that is when the number of discretization points is higher than the number of observations.

The last application deals with spectrometric data and allows us to show how a functional transformation (derivative calculation) can improve the efficiency of SVMs. For this application, we do not use the consistent methodology but a projection on a spline space that permits easy derivative calculations.

For reasons of simplicity, the parameter C is chosen from a finite set of values (in general less than 10 values) growing exponentially (for instance $0.1, 1, 10, \ldots$). In each simulation, the kernel family is fixed (e.g., Gaussian kernels). A finite set of fully specified candidate kernels are chosen in this family (for instance approximately 10 values of σ in the case of the Gaussian kernel family) and the best kernel is selected as described in the previous section.

6.1. *Speech recognition*

We first illustrate in this section the consistent learning procedure given in Section 5. We compare it to the original procedure based on k-nn described in [3]. In practice, the only difference between the approaches is that we use a SVM whereas [3] uses a k-nn.

The problems considered in [3] consist in classifying speech samples.[1] There are three problems with two classes each: classifying "yes" against "no", "boat" against "goat" and "sh" against "ao". For each problem, we have 100 functions. Table 1 gives the sizes of the classes for each problem.

Each function is described by a vector in $\mathbb{R}^{8192}$ which corresponds to a digitized speech frame. The goal of this benchmark is to compare data processing methods that make minimal assumptions on the data: no prior knowledge is used to preprocess the data.

In order to directly compare to results from [3], performances of the algorithms are assessed by a leave-one-out procedure: 99 functions are used as the learning set (to which the split sample procedure is applied to choose SVM) and the remaining function provides a test example. This process is repeated 100 times, each time using a different function as the test example.

[1]Data are available at http://www.math.univ-montp2.fr/~biau/bbwdata.tgz

Table 1
Sizes of the classes for the speech recognition problem

Problem	Class 1	Class −1
Yes/no	48	52
Boat/goat	55	45
sh/ao	42	58

Table 2
Error rate for reference methods for the speech recognition problem (leave-one out)

Problem	k-nn (%)	QDA (%)
Yes/no	10	7
Boat/goat	21	35
sh/ao	16	19

Table 3
Error rate for SVM based methods for the speech recognition problem (leave-one out)

Problem/ Kernel	Linear (direct) (%)	Linear (projection) (%)	Gaussian (projection) (%)
Yes/no	58	19	10
Boat/goat	46	29	8
sh/ao	47	25	12

While the procedure described in Section 5.2 allows us to choose most of the parameters, both the basis $\{\Psi_j\}_{j\geqslant 1}$ and the penalty term λ_d can be freely chosen. To focus on the improvement provided by SVM over k-nn, we have used the same elements as [3]. As the data are temporal patterns, [3] relies on the Fourier basis (moreover, the fast Fourier transform allows an efficient calculation of the coordinates of the data on the basis). The penalty term is 0 for all d below 100 and a high value (for instance 1000) for $d>100$. This allows to only evaluate the models for $d\leqslant 100$ because the high value of λ_d for higher d prevents the corresponding models to be chosen, regardless of their performances. As pointed out in [3], this choice appears to be safe as most of the dimensions then selected are much smaller than 50.

The last free parameter is the split between the training set and the validation set. As in [3] we have used the first 50 examples for training and the remaining 49 for validation. We report the error rate for each problem and several methods in Tables 2 and 3.

Tables 2 has been reproduced from [3]. QDA corresponds to quadratic discriminant analysis performed, as for k-nn, on the projection of the data onto a finite-dimensional subspace induced by the Fourier basis. Table 3 gives results obtained with SVMs. The second column, "linear (direct)", corresponds to the direct application of the procedure described in Section 3.1.2, without any prior projection. This is in fact the plain linear SVM directly applied to the original data. The two other columns corresponds to the SVM applied to the projected data, as described in Section 5.2.

The most obvious fact is that the plain linear kernel gives very poor performances, especially compared to the functional kernels on projections: its results are sometimes worse than the rule that affects any observation to the dominating class. This shows that the ridge regularization of problem (R_λ) is not well adapted to functional data, a fact that was already known in the context of linear discriminant analysis [17]. The projection operator improves the results of the linear kernel, but not enough to reach the performance levels of k-nn. It seems that the projected problem is therefore nonlinear.

As expected, the functional Gaussian SVM performs generally better than k-nn and QDA, but the training times of the methods are not comparable. On a mid range personal computer, the full leave-one-out evaluation procedure applied to Gaussian SVM takes approximately one and half hours (using LIBSVM [6] embedded in the package e1071 of the R software [28]), whereas the same procedure takes only a few minutes for k-nn and QDA.

The performances of SVM with Gaussian kernel directly used on the raw data (in $\mathbb{R}^{8192}$) are not reported here as they are quite meaningless. The results are indeed extremely sensitive to the way the grid search is conducted, especially for the value of C, the regularization parameter. On the "yes/no" data set for instance, if the search grid for C contains only values higher than 1, then the leave-one-out gives 19% of error. But in each case, the value $C=1$ is selected on the validation set. When the grid search is extended to smaller values, the smallest value is always selected and the error rate increases up to 46%. Similar behaviors occur for the other data sets. On this benchmark, the performances depend in fact on the choice of the search grid for C. This is neither the case of the linear kernel on raw data, nor the case for the projection based kernels. This is not very surprising as Gaussian kernels have some locality problems in very high-dimensional spaces (see [15]) that makes them difficult to use.

6.2. *Using wavelet basis*

In order to investigate the limitation of the direct use of the linear SVM, we have applied them to another speech recognition problem. We studied a part of TIMIT database which was used in [17].[2] The data are log-periodograms corresponding to recording phonemes of 32 ms duration (the length of each log-periodogram is 256). We have chosen to restrict ourselves to classifying "aa" against "ao", because this is the most difficult sub-problem in the database. The database is a multi-speaker database. There are 325 speakers in the training set and 112 in the test set. We have 519 examples for "aa" in the training set (759 for

[2]Data are available at http://www-stat.stanford.edu/~tibs/ElemStatLearn/datasets/phoneme.data

Table 4
Error rate for all methods on the test set

Functional Gaussian SVM (%)	Functional linear SVM (%)	Linear SVM(%)
22	19.4	20

"ao") and 176 in the test set (263 for "ao"). We use the split sample approach to choose the parameters on the training set (50% of the training examples are used for validation) and we report the classification error on the test set.

Here, we do not use a Fourier basis as the functions are already represented in a frequency form. As the data are very noisy, we decided to use a hierarchical wavelet basis (see e.g., [25]). We used the same penalty term as in Section 6.1. The error rate on the test set is reported in Table 4. It appears that functional kernels are not as useful here as in the previous example, as a linear SVM applied directly to the discretized functions (in $\mathbb{R}^{256}$) performs as well as a linear SVM on the wavelet coefficients. A natural explanation is that the actual dimension of the input space (256) is smaller than the number of training examples (639) which means that evaluating the optimal coefficients of the SVM is less difficult than in the previous example. Therefore, the additional regularization provided by reducing the dimension with a projection onto a small dimensional space is not really useful in this context.

6.3. Spectrometric data set

We study in this section spectrometric data from food industry.[3] Each observation is the near infrared absorbance spectrum of a meat sample (finely chopped), recorded on a Tecator Infratec Food and Feed Analyser (we have 215 spectra). More precisely, an observation consists in a 100 channel spectrum of absorbances in the wavelength range 850–1050 nm (see Fig. 1). The classification problem consists in separating meat samples with a high fat content (more than 20%) from samples with a low fat content (less than 20%).

It appears on Fig. 1 that high fat content spectra have sometimes two local maxima rather than one: we have therefore decided to focus on the curvature of the spectra, i.e., to use the second derivative. Fig. 2 shows that there is more differences between the second derivatives of each class than between the original curves.

The data set is split into 120 spectra for learning and 95 spectra for testing. The problem is used to compare standard kernels (linear and Gaussian kernels) to a derivative based kernel. We do not use the consistent procedure here as we choose a fixed spline subspace to represent the functions so as to calculate their second derivative. However, the parameters C and σ are still chosen by a split sample approach that divides the 120 learning samples into 60 spectra for learning and 60 spectra for validation. The dimension of the spline subspace is obtained thanks to a leave-one-out procedure applied to the whole set of input functions, without taking into account classes (see [33] for details).

[3]Data are available on statlib at http://lib.stat.cmu.edu/datasets/tecator

The performances depend of course on the random split between learning and test. We have therefore repeated this splitting 250 times (as we do not select an optimal projection dimension, the procedure is much faster than the one used for both previous experiments). Table 5 gives the mean error rate of those experiments on the test set.

The results show that the problem is less difficult that the previous ones. Nevertheless, it also appears that a functional transformation improves the results: the use of a Gaussian kernel on second derivatives gives significantly better results than the use of an usual kernel (linear or Gaussian) on the original data (t-test results). The relatively bad performances of the Gaussian kernel on plain data can be explained by the fact that a direct comparison of spectra based on their $L^2(\mu)$ norm is in general dominated by the mean value of those spectra which is not a good feature for classification in spectrometric problems. The linear kernel is less sensitive to this problem and is not really improved by the derivative operator. One possible interpretation of this behavior is that curvature can be estimated by linear combination of the original spectral variables with some finite difference calculations; the linear SVM might obtain good performances on the raw spectra thanks to this type of calculation. In the Gaussian case, the use of a functional transformation introduces expert knowledge (i.e., curvature is a good feature for some spectrometric problems) and allows to overcome most of the limitations of the original kernel.

7. Conclusion

In this paper, we have shown how to use support vector machines (SVMs) for functional data classification. While plain linear SVMs could be used directly on functional data, we have shown the benefits of using adapted functional kernels. We have defined projection based kernels that provide a consistent learning procedure for functional SVMs. We have also introduced transformation based kernels that allow to take into account expert knowledge (such as the fact that the curvature of a function can be more discriminant than its values in some applications). Both types of kernels have been tested on real world problems. The experiments gave very satisfactory results and showed that for some types of functional data, the performances of SVM based classification can be improved by using kernels that make use of the functional nature of the data.

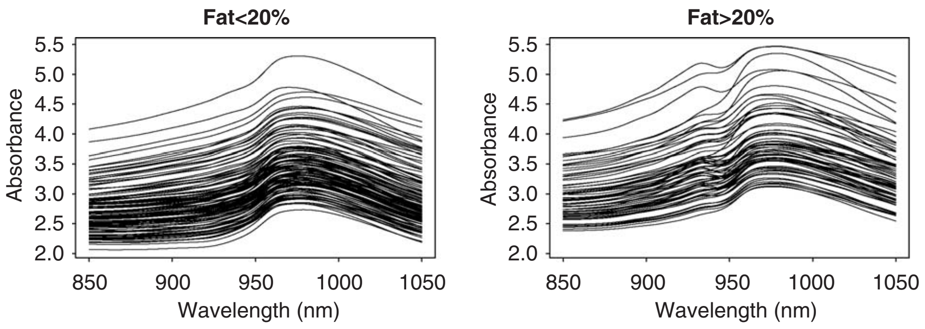

Fig. 1. Spectra for both classes.

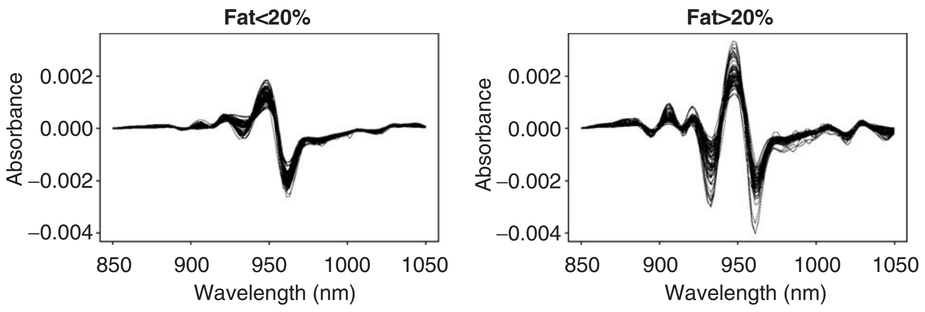

Fig. 2. Second derivatives of the spectra for both classes.

Table 5
Mean test error rate for all methods on the spectrometric dataset

Kernel	Mean test error (%)
Linear	3.38
Linear on second derivatives	3.28
Gaussian	7.5
Gaussian on second derivatives	2.6

Acknowledgements

The authors thank the anonymous referees and the editor for their valuable suggestions that helped to improve this paper.

Appendix A. Proofs

In order to simplify the notation, we denote $l = l_N$ when N is obvious. We also denote $X^{(d)} = P_{V_d}(X)$ and $x_i^{(d)} = P_{V_d}(x_i)$.

The proof of the consistency result of [3] is based on an oracle. We demonstrate a similar inequality: for N large enough,

$$L(f_{a^*}) - L^* \leqslant \inf_{d\geqslant 1}\left[L_d^* - L^* + \inf_{C\in\mathscr{I}_d,\ K\in\mathscr{J}_d}(L(f_a) - L_d^*) + \frac{\lambda_d}{\sqrt{m}}\right] + \sqrt{\frac{32(l+1)\log m}{m}} + 128\Delta\sqrt{\frac{1}{32m(l+1)\log m}}, \tag{A.1}$$

where $m = N - l$, $\Delta \equiv \sum_{d\geqslant 1}|\mathscr{J}_d|e^{-\lambda_d^2/32} < +\infty$ and L_d^* is the Bayes error for the projected problem, i.e. $L_d^* = \inf_{f:\mathbb{R}^d\to\{-1;1\}}\mathbb{P}(f(X^{(d)})\neq Y)$.

Following [3], we see that the definition of $a^* = (d^*, K^*, C^*)$ leads to,

$$\widehat{L}(f_{a^*}) + \frac{\lambda_{d^*}}{\sqrt{m}} \leqslant \widehat{L}(f_a) + \frac{\lambda_d}{\sqrt{m}}$$

for all $a = (d, C, K)$ in $\mathscr{A} = \bigcup_{d\geqslant 1}\{d\}\times\mathscr{J}_d\times[0,\mathscr{C}_d]$. Then, for all $\varepsilon > 0$,

$$\begin{aligned}\mathbb{P}\left(L(f_{a^*}) - \widehat{L}(f_a) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right) &\leqslant \mathbb{P}\left(L(f_{a^*}) - \widehat{L}(f_{a^*}) > \frac{\lambda_{d^*}}{\sqrt{m}} + \varepsilon\right)\\ &\leqslant \sum_{d\geqslant 1}\mathbb{P}\left(L(f_{(d,C^*,K^*)}) - \widehat{L}(f_{(d,C^*,K^*)}) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right)\\ &\leqslant \sum_{d\geqslant 1,\ K\in\mathscr{J}_d}\mathbb{P}\left(L(f_{(d,C^*,K)}) - \widehat{L}(f_{(d,C^*,K)}) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right).\end{aligned} \tag{A.2}$$

In [3], the right-hand side of the inequality is bounded by the use of the union bound on $\mathscr{A}$. Here, $[0,\mathscr{C}_d]$ is not countable and thus we cannot do the same. We will then use the generalization capability of a set of linear classifiers via its shatter coefficient. Actually, when d and K are set, $f_{(d,C^*,K)}$ is an affine discrimination function built from the observation projections and the kernel K. More precisely, we have:

$$\text{for all } x \text{ in } \mathcal{X}, \quad f_a(x^{(d)}) = \sum_{n=1}^{l} \alpha_n^* y_n K(x_n^{(d)}, x^{(d)}) + b^*.$$

Then, f_a has the form $b+f$ where f is chosen in the set of functions spanned by $\{K(x_1^{(d)},.),\ldots,K(x_l^{(d)},.)\}$. Let us denote by $\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})$ this set of classifiers and, for all f in $\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})$, we introduce $L^l(f) = \mathbb{P}(f(X^{(d)}) \neq Y|(x_1,y_1),\ldots,(x_l,y_l))$. By Theorem 12.6 in [11], we then have, for all $v>0$,

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})} |\widehat{L}(f) - L^l(f)| > v \,\middle|\, (x_1,y_1),\ldots,(x_l,y_l)\right)$$
$$\leqslant 8\mathcal{S}(\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)}),m)\mathrm{e}^{-mv^2/32},$$

where $\mathcal{S}(\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)}),m)$ is the shatter coefficient of $\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})$, that is the maximum number of different subsets of m points that can be separated by the set of classifiers $\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})$. This set is a vector space of dimension less or equal to $l+1$, therefore according to [11, Chapter 13], $\mathcal{S}(\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)}),m) \leqslant m^{l+1}$. This implies that, for all $(d,K) \in \mathbb{N}^* \times \mathcal{J}_d$,

$$\mathbb{P}\left(L(f_{(d,C^*,K)}) - \widehat{L}(f_{(d,C^*,K)}) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right)$$
$$= \mathbb{E}\left[\mathbb{P}\left(L(f_{(d,C^*,K)}) - \widehat{L}(f_{(d,C^*,K)}) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon \,\middle|\, (x_1,y_1),\ldots,(x_l,y_l)\right)\right]$$
$$\leqslant \mathbb{E}\left[\mathbb{P}\left(\sup_{f\in\mathcal{F}_K(x_1^{(d)},\ldots,x_l^{(d)})} |\widehat{L}(f) - L^l(f)| > \frac{\lambda_d}{\sqrt{m}} + \varepsilon \,\middle|\, (x_1,y_1),\ldots,(x_l,y_l)\right)\right]$$
$$\leqslant 8m^{l+1}\mathrm{e}^{-\lambda_d^2/32}\mathrm{e}^{-m\varepsilon^2/32}. \tag{A.3}$$

Combining (A.2) and (A.3), we finally see that

$$\mathbb{P}\left(L(f_{a^*}) - \widehat{L}(f_a) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right) \leqslant 8\Delta m^{l+1}\mathrm{e}^{-m\varepsilon^2/32}.$$

If Z is a positive random variable, we have obviously

$$\mathbb{E}(Z) \leqslant \mathbb{E}(Z\mathbb{1}_{\{Z>0\}}) = \int_0^{+\infty} \mathbb{P}(Z \geqslant \varepsilon)\,\mathrm{d}\varepsilon.$$

For $Z = L(f_{a^*}) - \widehat{L}(f_a) - \lambda_d/\sqrt{m}$, this leads, for all a in $\bigcup_d \{d\} \times \mathcal{J}_d \times \mathcal{J}_d$, to

$$L(f_{a^*}) \leqslant \mathbb{E}(\widehat{L}(f_a)) + \frac{\lambda_d}{\sqrt{m}} + \int_0^{+\infty} \mathbb{P}\left(L(f_{a^*}) - \widehat{L}(f_a) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right)\mathrm{d}\varepsilon.$$

Finally, following [3], for all $u>0$,

$$\int_0^{+\infty} \mathbb{P}\left(L(f_{a^*}) - \widehat{L}(f_a) > \frac{\lambda_d}{\sqrt{m}} + \varepsilon\right)\mathrm{d}\varepsilon$$
$$\leqslant \int_0^u 1\,\mathrm{d}\varepsilon + \int_u^{+\infty} 8\Delta m^{l+1}\mathrm{e}^{-m\varepsilon^2/32}\,\mathrm{d}\varepsilon$$
$$\leqslant u + 128\Delta m^{l+1}\int_u^{+\infty}\left(\frac{1}{16} + \frac{1}{m\varepsilon^2}\right)\mathrm{e}^{-m\varepsilon^2/32}\,\mathrm{d}\varepsilon$$

and then

$$L(f_{a^*}) \leqslant \mathbb{E}(\widehat{L}(f_a)) + \frac{\lambda_d}{\sqrt{m}} + u + \frac{128\Delta m^l}{u}\mathrm{e}^{-mu^2/32};$$

if we set $u = \sqrt{32(l+1)\log m/m}$ and by the equality $\mathbb{E}(\widehat{L}(f_a)) = L(f_a)$, we deduce that, for all a in $\mathcal{A}$,

$$L(f_{a^*}) \leqslant L(f_a) + \frac{\lambda_d}{\sqrt{m}} + \sqrt{\frac{32(l+1)\log m}{m}} + 128\Delta\sqrt{\frac{1}{32(l+1)\log m}}$$

which finally proves oracle (A.1).

We conclude via the following steps:

(1) $\lim_{m\to+\infty}\sqrt{\frac{32(l+1)\log m}{m}} + 128\Delta\sqrt{\frac{1}{32m(l+1)\log m}} = 0$ from the assumptions of Theorem 1;
(2) Lemma 5 in [3] shows that $L_d^* - L^* \overset{d\to+\infty}{\longrightarrow} 0$;
(3) Let $\varepsilon>0$. If we take a d_0 such that, for all $d \geqslant d_0$, $L_d^* - L^* \leqslant \varepsilon$. To conclude, we finally have to prove that $\inf_{(C,K)\in\mathcal{J}_{d_0}\times\mathcal{J}_{d_0}} L(f_{(d_0,C,K)}) - L_{d_0}^* \overset{N\to+\infty}{\longrightarrow} 0$. This is a direct consequence of Theorem 2 in [40]. Let us show that the hypotheses of this theorem are fulfilled:

(a) Theorem 2 in [40] is valid for universal kernels that satisfy some requirements on their covering numbers.
As we focus on $\inf_{(C,K)\in\mathcal{J}_{d_0}\times\mathcal{J}_{d_0}} L(f_{(d_0,C,K)})$, we can choose freely the kernel and the regularization parameter in $\mathcal{J}_{d_0}\times\mathcal{J}_{d_0}$. Therefore, we choose K_{d_0} an universal kernel with covering number of the form $\mathcal{O}(\varepsilon^{-v_{d_0}})$ for some $v_{d_0}>0$ (this is possible according to our hypotheses).
(b) Theorem 2 in [40] asks for $X^{(d)}$ to take its values in a compact set of $\mathbb{R}^d$.
Actually, X is bounded in $\mathcal{X}$ so, by definition of $x \to x^{(d)}$, $X^{(d)}$ takes its values in a bounded set of $\mathbb{R}^d$ which is included in a compact set of $\mathbb{R}^d$;
(c) Finally, Theorem 2 in [40] requests a particular behavior for C_l, the regularization parameter used for l examples: C_l is such that $lC_l \to +\infty$ and $C_l = \mathcal{O}(l^{\beta-1})$ for some $0<\beta<\frac{1}{v_{d_0}}$.

Let β_{d_0} be any number in $]0, \frac{1}{v_{d_0}} \wedge 1[$ (where $a \wedge b$ denotes the infimum between a and b). Then, let C_l be $l^{\beta_{d_0}-1}$. This defines a sequence of real numbers included in $]0, 1[$ which fulfills the requirements stated above. As $\mathscr{C}_{d_0} \geqslant 1$ for all $l \geqslant 2$, we have $C_l \in [0, \mathscr{C}_{d_0}]$ therefore such choice of the regularization parameters is compatible with the hypothesis of our theorem.

This allows to apply Theorem 2 in [40] which implies that $L(f_{(d_0,(C_l),K_{d_0})})$ converges to $L^*_{d_0}$ and finally to obtain the conclusion.

References

[1] N. Aronszajn, Theory of reproducing kernels, Trans. Am. Math. Soc. 68 (3) (1950) 337–404.

[2] P. Besse, J. Ramsay, Principal component analysis of sampled curves, Psychometrica 51 (1986) 285–311.

[3] G. Biau, F. Bunea, M. Wegkamp, Functional classification in Hilbert spaces, IEEE Trans. Inf. Theory 51 (2005) 2163–2172.

[4] H. Cardot, F. Ferraty, P. Sarda, Functional linear model, Stat. Probab. Lett. 45 (1999) 11–22.

[5] H. Cardot, F. Ferraty, P. Sarda, Spline estimators for the functional linear model, Statistica Sinica 13 (2003) 571–591.

[6] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[7] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines, Cambridge University Press, Cambridge, UK, 2000.

[8] J. Dauxois, A. Pousse, Les analyses factorielles en calcul des probabilités et en statistiques: essai d'étude synthétique, Thèse d'état, Université Paul Sabatier, Toulouse, 1976.

[9] J. Dauxois, A. Pousse, Y. Romain, Asymptotic theory for the principal component analysis of a vector of random function: some applications to statistical inference, J. Multivar. Anal. 12 (1982) 136–154.

[10] J. Deville, Méthodes statistiques et numériques de l'analyse harmonique, Ann. l'INSEE 15 (1974) 3–97.

[11] L. Devroye, L. Györfi, G. Lugosi (Eds.), A Probabilistic Theory of Pattern Recognition, vol. 21, Applications of Mathematics, Springer, Berlin, 1996.

[12] T. Evgeniou, M. Pontil, T. Poggio, Regularization networks and support vector machines, Adv. Comput. Math. 13 (1) (2000) 1–50.

[13] L. Ferré, N. Villa, Multi-layer perceptron with functional inputs: an inverse regression approach, Scand. J. Stat., 2006, to be published.

[14] L. Ferré, N. Villa, Discrimination de courbes par régression inverse fonctionnelle, Rev. Stat. Appl. 1 (LIII) (2005) 39–57.

[15] D. Francois, V. Wertz, M. Verleysen, About the locality of kernels in high-dimensional spaces, in: ASMDA 2005, International Symposium on Applied Stochastic Models and Data Analysis, Brest, France, 2005, pp. 238–245.

[16] I. Frank, J.H. Friedman, A statistical view of some chemometrics regression tools, Technometrics 35 (1993) 109–148.

[17] T. Hastie, A. Buja, R. Tibshirani, Penalized discriminant analysis, Ann. Stat. 23 (1995) 73–102.

[18] T. Hastie, C. Mallows, A discussion of A statistical view of some chemometrics regression tools by I.E. Frank and J.H. Friedman, Technometrics 35 (1993) 140–143.

[19] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, The entire regularization path for the support vector machine, J. Mach. Learn. Res. 5 (2004) 1391–1415.

[20] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, Berlin, 2001.

[21] A.E. Hoerl, R.W. Kennard, Ridge regression: biased estimation for nonorthogonal problems, Technometrics 12 (1) (1970) 55–67.

[22] G.M. James, T.J. Hastie, Functional linear discriminant analysis for irregularly sampled curves, J. R. Stat. Soc. Ser. B 63 (2001) 533–550.

[23] S. Leurgans, R. Moyeed, B. Silverman, Canonical correlation analysis when the data are curves, J. R. Stat. Soc. B 55 (3) (1993) 725–740.

[24] C.-J. Lin, Formulations of support vector machines: a note from an optimization point of view, Neural Comput. 2 (13) (2001) 307–317.

[25] S. Mallat, Multiresolution approximation and wavelet orthonormal bases of l2, Trans. Am. Math. Soc. 315 (1989) 69–87.

[26] B.D. Marx, P.H. Eilers, Generalized linear regression on sampled signals with penalized likelihood, in: R.H.A. Forcina, G.M. Marchetti, G. Galmacci (Eds.), Statistical Modelling. Proceedings of the 11th International workshop on Statistical Modelling, Orvietto, 1996.

[27] S. Pezzulli, B. Silverman, On smoothed principal components analysis, Comput. Stat. 8 (1993) 1–16.

[28] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0.

[29] J. Ramsay, B. Silverman, Functional Data Analysis, Springer Series in Statistics, Springer, Berlin, 1997.

[30] J.O. Ramsay, C.J. Dalzell, Some tools for functional data analysis (with discussion), J. R. Stat. Soc. Ser. B 53 (1991) 539–572.

[31] F. Rossi, B. Conan-Guez, Functional multi-layer perceptron: a nonlinear tool for functional data analysis, Neural Networks 18 (1) (2005) 45–60.

[32] F. Rossi, B. Conan-Guez, A. El Golli, Clustering functional data with the SOM algorithm, in: Proceedings of ESANN 2004, Bruges, Belgium, 2004, pp. 305–312.

[33] F. Rossi, N. Delannay, B. Conan-Guez, M. Verleysen, Representation of functional data in neural networks, Neurocomputing 64 (2005) 183–210.

[34] F. Rossi, N. Villa, Classification in Hilbert spaces with support vector machines, in: ASMDA 2005, International Symposium on Applied Stochastic Models and Data Analysis, Brest, France, 2005, pp. 635–642.

[35] I.W. Sandberg, Notes on weighted norms and network approximation of functionals, IEEE Trans. Circuits Syst.–I: Fundam. Theory Appl. 43 (7) (1996) 600–601.

[36] I.W. Sandberg, L. Xu, Network approximation of input–output maps and functionals, Circuits Systems Signal Process. 15 (6) (1996) 711–725.

[37] A. Smola, B. Schölkopf, On a kernel-based method for pattern recognition, regression, approximation and operator inversion, Algorithmica 22 (10) (1998) 211–231.

[38] A. Smola, B. Schölkopf, K.-R. Müller, The connection between regularization operators and support vector kernels, Neural Networks 11 (1998) 637–649.

[39] I. Steinwart, On the influence of the kernel on the consistency of support vector machines, J. Mach. Learn. Res. 2 (2001) 67–93.

[40] I. Steinwart, Support vector machines are universally consistent, J. Complexity 18 (3) (2002) 768–791.

[41] M.B. Stinchcombe, Neural network approximation of continuous functionals and continuous functions on compactifications, Neural Networks 12 (3) (1999) 467–477.

[42] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 1995.

[43] J. Vert, K. Tsuda, B. Schölkopf, in: A Primer on Kernel Methods, MIT Press, Cambridge, MA, 2004, pp. 35–70 (Chapter 2).

[44] N. Villa, F. Rossi, Support vector machine for functional data classification, in: Proceedings of ESANN 2005, Bruges, Belgium, Avon Books, New York, 2005, pp. 467–472.

Fabrice Rossi was born in 1971 in France. He is a former student of the E.N.S. (Ecole Normale Supérieure de la rue d'Ulm). He received a Ph.D. degree in applied mathematics from the Université Paris-IX Dauphine in 1996. He is presently research scientist in the AxIS Project Team at INRIA (Institut National de Recherche en Informatique et en Automatique). From 1997 to 2003, he was also Assistant Professor in computer science and applied mathematics at the Université Paris-IX Dauphine. He is member of the CEREMADE, a joint research group from the Université Paris-IX Dauphine and the French C.N.R.S. (Centre National de la Recherche Scientifique). His research activities focus on artificial neural networks and nonlinear data analysis methods.

Nathalie Villa was born in Tulle (France) in 1976. She received the french "Agrégation" in mathematics in 1999 and the Ph.D. degree in applied mathematics in 2005. She is a lecturer of the University Toulouse Le Mirail since 2000 and a member of the GRIMM research team in this university. Her research activities focus on functional data analysis by nonlinear methods such as multilayer perceptron or SVM and on functional inverse regression models.