Multicategory SVMs by Minimizing the Distances Among Convex-Hull Prototypes

Ricardo Ñanculef, Carlos Concha, Héctor Allende, Diego Candel
Federico Santa María University, CP 110-V Valparaíso, Chile
{jnancu,cconcha,hallende,dcandel}@inf.utfsm.cl

Claudio Moraga
European Centre for Soft Computing 33600 Mieres, Asturias, Spain
Technical University of Dortmund, 44221 Dortmund, Germany
mail@claudio-moraga.eu

Abstract

In this paper, we study a single objective extension of support vector machines for multicategory classification. Extending the dual formulation of binary SVMs, the algorithm looks for minimizing the sum of all the pairwise distances among a set of prototypes, each one constrained to one of the convex-hulls enclosing a class of examples. The final discriminant system is built looking for an appropriate reference point in the feature space. The obtained method preserves the form and complexity of the binary case, optimizing just one convex objective function with m variables and $2m+K$ constraints, where m is the number of examples and K the number of classes. Non-linear extensions are straightforward using kernels while "soft margin versions" can be obtained by using reduced convex hulls. Experimental results in well-known UCI benchmarks are presented, comparing the accuracy and efficiency of the proposed approach with other state-of-the-art methods.

1 Introduction

Support Vector Machines (SVMs) constitute one of the standard tools for machine learning and data mining, successfully applied to a variety of real-world problems from different fields. SVMs were originally formulated for binary classification [8]. In many problems however, observations can belong to more than two classes, which makes the extension of SVMs to multicategory classification an issue of primary interest in practice.

As it can be seen in [4], there are two types of formulations for multicategory SVMs. One corresponds to using several binary classifiers, separately trained and joined into a multicategory decision function. Other type of formulation for multiclass SVMs consists in reformulating the large margin problem to directly address the multicategory case in just one optimization function. In [4], the study concluded that single objective methods are less suitable for practical use due to the greater number of variables they have to consider simultaneously: an order of $K \cdot m$ (primal or dual) variables where K is the number of classes and m the number of examples.

In this paper we propose a single objective method to extend binary SVMs to the multicategory case. The classifier is obtained by looking for a set of prototypes, each one associated to one of the available classes, and a reference point in the feature space from which a discriminant system will be built in order to distinguish new patterns. Extending the dual formulation of binary SVMs developed in [1], the algorithm looks for minimizing the sum of all the pairwise distances among the set of prototypes which are explicitly constrained to the convex-hulls enclosing each class of examples. The main appeal of the resulting algorithm is that the optimization problem has m variables and $2m + K$ constraints and is hence considerably lighter than other single objective implementations in which the number of constraints or variables is proportional to $K \cdot m$. The idea of minimizing the sum of all pairwise distances between prototypes has been already exploited in [3]. In our approach however, instead of an explicit error minimization, we restrict the prototypes to the convex-hulls enclosing the class they represent and look for an appropriate reference point to define the discriminant system.

The rest of this paper is organized as follows. In the next section, a brief overview of the dual perspective of SVMs [1] is presented. In section 3 we present and analyze our method to extend this formulation to the multicategory case. Non-linear and "soft margin" versions of the classifier are

978-0-7695-3326-1/08 $25.00 © 2008 IEEE
DOI 10.1109/HIS.2008.173

then provided. Finally we present a set of experiments in well-known UCI benchmarks, comparing our technique with other state-of-the-art methods, discussing the conclusions of this work.

2 Binary Support Vector Machines

Suppose we are given with a finite set of examples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ in a feature space $\mathcal{X} \subset \mathbb{R}^n$, some of which belong to a class $\mathcal{C}_1$ and the others to a class $\mathcal{C}_2$, and we are asked to learn a decision function $f(\mathbf{x}) : \mathcal{X} \rightarrow \{1, 2\}$ to distinguish patterns of class $\mathcal{C}_1$ from patterns of the other class. A way to accomplish this task is by modeling the boundary between $\mathcal{C}_1$ and $\mathcal{C}_2$ using a separating hyperplane $\mathcal{H} = \{\mathbf{x} \in \mathcal{X} : \mathbf{w}^T\mathbf{x} - b = 0\}$ with parameters $\mathbf{w} \in \mathcal{X}$ $(\neq 0)$ and $b \in \mathbb{R}$. When the problem is linearly separable [8], an infinite number of such separating hyperplanes can be found. From the set of possible solutions, SVMs choose the one maximizing the so called *margin*, defined as the distance to the closest point from either class. If we label examples $\mathbf{x}_i \in \mathcal{C}_1$ with $y_i = +1$ and examples $\mathbf{x}_i \in \mathcal{C}_2$ with -1, the hyperplane maximizing the margin can be found by solving

$$\begin{aligned} \textbf{minimize}_{\{\mathbf{w},b\}} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \textbf{s.t.} \quad & y_i\left(\mathbf{w}^T\mathbf{x} - b\right) \geq 1 \;\; i \in I \end{aligned} \tag{1}$$

where $I = \{1, 2, \ldots, m\}$. As shown in [1], an equivalent geometric formulation of the maximum margin problem (1) can be obtained by defining the convex hulls H_1, H_2 enclosing each class and looking for the points $\mathbf{v}_1 \in H_1$ and $\mathbf{v}_2 \in H_2$ minimizing $\frac{1}{2}\|\mathbf{v}_1 - \mathbf{v}_2\|^2$. The maximum margin hyperplane is the one which orthogonally bisects the line $\mathbf{v}_1 - \mathbf{v}_2$, that is, the solution $(\mathbf{w}, b)$ of (1) corresponds to choose $\mathbf{w} = \mathbf{v}_1 - \mathbf{v}_2$ and $b = (\mathbf{v}_1 + \mathbf{v}_2)^T(\mathbf{v}_1 - \mathbf{v}_2)/2$. Since any point $x \in H_i$ can be written as a convex combination of the examples of class $\mathcal{C}_i$, this geometric formulation leads to the following optimization problem in terms of a parameter vector $\mathbf{u} \in \mathbb{R}^m$

$$\begin{aligned} \textbf{minimize}_{\{\mathbf{u}\}} \quad & \frac{1}{2}\|\sum_{i:\mathbf{x}_i \in \mathcal{C}_1} u_i\mathbf{x}_i - \sum_{i:\mathbf{x}_i \in \mathcal{C}_2} u_i\mathbf{x}_i\|^2 \\ \textbf{s.t.} \quad & \sum_{i \in I_1} u_i = 1, \;\; \sum_{i \in I_2} u_i = 1, \;\; u_i \geq 0 \end{aligned} \tag{2}$$

where $I_1 = \{i : \mathbf{x}_i \in \mathcal{C}_1\}$, $I_2 = \{i : \mathbf{x}_i \in \mathcal{C}_2\}$. Since class $\mathcal{C}_1$ is identified with a label $+1$ and $\mathcal{C}_2$ with a label -1, classification of new patterns $\mathbf{x}$ can be carried out using the decision function $f(\mathbf{x}) = \text{sign}\left(\mathbf{w}^T\mathbf{x} - b\right)$. Note that the latter criterion is equivalent to classify $\mathbf{x}$ as $\mathcal{C}_1$ if $(\mathbf{x} - c)^T(\mathbf{v}_1 - c) > (\mathbf{x} - c)^T(\mathbf{v}_2 - c)$ and as $\mathcal{C}_2$ otherwise, where $c = (\mathbf{v}_1 + \mathbf{v}_2)/2$ is the center of the configuration defined by $\mathbf{v}_1$ and $\mathbf{v}_2$.

The extension of SVMs to the case of non-linearly separable classes usually proceeds by introducing slack variables on problem (1). In terms of the dual formulation, however, the problem with inseparable classes is that the convex hulls of both will intersect. A way to deal with this case consists in bounding the potential influence of each example in the convex combination, which has the effect of comprising or reducing the original convex hulls until we get a separable configuration. Using these reduced convex-hulls, the soft-margin version of (2) is hence obtained by replacing the constraint $u_i \geq 0$ in problem (2) by $\eta \geq u_i \geq 0$, where parameter η has a regularization effect.

Extension of the above formulation to non-linear decision functions proceeds by replacing dot products $\mathbf{x}_i^T\mathbf{x}_j$ in the original space $\mathcal{X}$ by a Mercer kernel $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ which corresponds to a dot product in a higher dimensional space $\Phi(\mathcal{X})$. Let us note that the objective function of (2) can be written explicitly in terms of dot products between patterns

$$\frac{1}{2}\|\sum_{i \in I_1} u_i\mathbf{x}_i - \sum_{i \in I_2} u_i\mathbf{x}_i\|^2 \;\; = \;\; \sum_{i,j} u_i u_j \mathbf{x}_i^T\mathbf{x}_j \tag{3}$$

Analogously, the components $\mathbf{w}^T\mathbf{x}$ and b in the decision function can be analogously re-written using dot products between training examples

$$\mathbf{w}^T\mathbf{x} = \sum_{i \in I_1} u_i\mathbf{x}_i^T\mathbf{x} - \sum_{i \in I_2} u_i\mathbf{x}_i^T\mathbf{x} \tag{4}$$

$$b = \sum_{i,j \in I_2} u_i u_j \mathbf{x}_i^T\mathbf{x}_j - \sum_{i,j \in I_1} u_i u_j \mathbf{x}_i^T\mathbf{x}_j \tag{5}$$

3 Minimizing the Distances Among Convex-Hull Prototypes

Let $\mathcal{X}$ be a subset of $\mathbb{R}^n$ and $\mathcal{Y}$ be a set of K possible labels representing different classes $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K$. Let $S = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m\}$ be a set of labeled examples $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y}$. The objective in multicategory classification consists in learning a decision function $f(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{Y}$ capable to predict the classes corresponding to new patterns $\mathbf{x} \in \mathcal{X}$. A simple way to approach this problem consists in finding a set of prototypes $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K \in \mathcal{X}$ each one associated to one of the possible classes, to implement a decision function of the form

$$f(\mathbf{x}) \;\; = \;\; \arg\max_i s(\mathbf{v}_i, \mathbf{x}) \tag{6}$$

where $s : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ measures the similarity between patterns in $\mathcal{X}$. This is actually the form of the classifier developed in [3], where $s(\mathbf{v}_i, \mathbf{x})$ is implemented using dot products $\mathbf{v}_i^T\mathbf{x}$. The existence of a set of prototypes capable to correctly classify the training data according to this criterion cannot in general be guaranteed. For example, if

we have two collinear classes of points at the same side of the origin, there does not exist a set of prototypes which can correctly distinguish the classes. A separability criterion needs hence to be introduced, which can then be relaxed, incorporating slack variables and non-linear kernels.
In this paper we work with a slightly different decision function of the form

$$f(\mathbf{x}) \quad = \quad \arg\max_i (\mathbf{v}_i - \mathbf{c})^T(\mathbf{x} - \mathbf{c}) \tag{7}$$

where $\mathbf{c}$ is a reference point in $\mathcal{X}$ from which we build the discriminant system, used to classify new patterns. Note that this is actually the form of the binary SVM as formulated in the previous section, in which the maximum margin classifier is built by selecting two prototypes from the convex-hulls enclosing each class of examples. Extending this construction we propose to restrict the selection of the prototypes $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K$ to the convex-hulls enclosing each one of the available classes. Such convex-hulls correspond to the minimal convex regions containing all the examples of a given category. By simplicity of notation, examples corresponding to a class $\mathcal{C}_k$ can be arranged as the rows of a matrix $\mathbf{X}_k \in M(m_k \times n)$, where m_k is the number of examples labeled as $\mathcal{C}_k$. The convex hull H_k associated to the class $k \in \{1, \ldots, K\}$ is hence defined as

$$\begin{aligned} H_k = \quad & \{\mathbf{x} : \mathbf{x} = \mathbf{X}_k^T \mathbf{u} \text{ for some } \mathbf{u} \in \mathbf{R}^{m_k} \text{ with} \\ & \mathbf{1}^T \mathbf{u} = 1 \text{ and } \mathbf{u} > 0\} \end{aligned} \tag{8}$$

In the binary case, $\mathbf{v}_1$ and $\mathbf{v}_2$ were selected as the points minimizing the distance between H_1 and H_2, which corresponds to solve

$$\begin{aligned} \textbf{minimize}_{\{\mathbf{v}_1, \mathbf{v}_2\}} \quad & \rho(\mathbf{v}_1, \mathbf{v}_2) = \frac{1}{2}\|\mathbf{v}_1 - \mathbf{v}_2\|^2 \\ \textbf{s.t.} \quad & \mathbf{v}_1 \in H_1 \;\; \mathbf{v}_2 \in H_2 \end{aligned} \tag{9}$$

A natural extension of this criterion corresponds to choose $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K$ as close as possible according to some operator $\rho : \bigtimes_{i=1}^{K} \mathcal{X} \rightarrow \mathbb{R}$ acting on K feature vectors simultaneously. We propose to consider the sum of all the pairwise distances among the convex-hull prototypes, that is the norms $\|\mathbf{v}_m - \mathbf{v}_l\|^2$ considering the $K \cdot (K-1)$ different pairs $\mathbf{v}_m, \mathbf{v}_l$. The prototypes selected according to this criterion are expected to characterize the configuration of the classes in the region where they confuse more, that is the boundary between the classes. Note that when we focus on a pair $\mathbf{v}_m, \mathbf{v}_l$ we are actually implementing a binary SVM criterion to find the prototypes which allows to define the boundary between the classes m and l. Similarly, a *one-versus-one* approach to implement multicategory SVMs builds $K \cdot (K-1)$ binary SVMs for each pair of classes m, l and then join the resulting classifiers. Hence, if we had a different prototype $\mathbf{v}_m$ for each pair of classes m, l we would have a single-objective implementation of the one-versus-one scheme. This would require a set of $K \cdot (K-1)$ prototypes and then the optimization problem would have at least $m \cdot (K-1)$ variables. In the proposed approach however we are using just one convex-hull prototype for each class which is optimized with respect to all the other classes simultaneously. This allows us to obtain the following optimization problem

$$\begin{aligned} \textbf{minimize}_{\{\mathbf{v}_i\}_{i=1}^K} \quad & \frac{1}{2} \sum_l \sum_{m<l} \|\mathbf{v}_m - \mathbf{v}_l\|^2 \\ \textbf{s.t.} \quad & \mathbf{v}_i \in H_i \; i = 1, \ldots, K \end{aligned} \tag{10}$$

To correctly classify the training data, the set of prototypes $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K$ must satisfy that $(\mathbf{v}_k - \mathbf{c})^T(\mathbf{x} - \mathbf{c}) > (\mathbf{v}_j - \mathbf{c})^T(\mathbf{x} - \mathbf{c})$ for all $\mathbf{x}$ belonging to the class i. We would like to note that in contrast to other multicategory SVMs implementations, we don't incorporate any explicit constraint to avoid or minimize the training error, which allows us to obtain a smaller optimization problem. Similarly, the dual construction of the binary SVM presented in the previous section does not include such constraints since the linear separability assumption and the restriction of the prototypes to the convex-hulls guarantee correct classification of the training data. A (sufficient but not necessary) separability condition for the multicategory case, is that for each class k the minimum inner product between any two examples $\mathbf{x}, \mathbf{y}$ of this class, both measured with respect to the reference point $\mathbf{c}$, be greater than the inner product with any point $\mathbf{z}$ of other class $j \neq k$, also measured with respect to $\mathbf{c}$. Let I_k be the set of examples of class C_k, $u_{s,r}$ the r-th component of $\mathbf{u}_s$ and $\mathbf{x}_{s,r}$ the r-th example of the class s. Since $\mathbf{v}_k = \mathbf{X}_k^T \mathbf{u}_k$ we have for all $\mathbf{x}_i$ such that $y_i = \mathcal{C}_k$

$$\begin{aligned} (\mathbf{v}_k - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) &= \sum_{s=1}^{m_k} u_{k,s}(\mathbf{x}_{k,s} - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) \\ &\geq \arg\min_{s \in I_k} (\mathbf{x}_{k,s} - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) \\ &> \arg\max_{j \neq s, s \in I_j} (\mathbf{x}_s - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) \\ \geq \sum_{s=1}^{m_j} u_{j,s}(\mathbf{x}_{j,s} - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) &= (\mathbf{v}_j - \mathbf{c})^T(\mathbf{x}_i - \mathbf{c}) \end{aligned} \tag{11}$$

as desired. The latter separability condition seems difficult to achieve in the original feature space $\mathcal{X}$. However, in the next section we introduce non-linear kernels and consider reduced convex-hulls instead of the raw convex-hulls which can make the configurations associated with each class compact enough to be separable from the rest of the classes using our classifier. Note now that a key component of our decision function is the reference point $\mathbf{c}$ from which we define the discriminant system. This has not been yet defined. We exploit this additional degree of freedom looking for a $\mathbf{c}$ such that the dot products between two different

discriminant vectors $(\mathbf{v}_i - \mathbf{c})$ and $(\mathbf{v}_j - \mathbf{c})$ be as negative as possible in order to avoid classification error. The point $\mathbf{c}$ is then obtained minimizing

$$D = \sum_l \sum_{m \neq i} (\mathbf{v}_l - \mathbf{c})^T (\mathbf{v}_m - \mathbf{c}) \tag{12}$$

Taking first and second derivatives in $\mathbf{c}$

$$\frac{\delta D}{\delta \mathbf{c}} = -2(K-1)\left(\sum_l \mathbf{v}_l\right) + 2K(K-1)\mathbf{c} \tag{13}$$

$$\frac{\delta D}{\delta \mathbf{c}} = 2K(K-1) > 0$$

which shows that the minimum is achieved with $\mathbf{c} = \sum_k \mathbf{v}_k / K$, that is the geometric center of the configuration given by the prototype vectors. It is interesting to note however that the selection of the prototypes is independent of the center defined by them and thus other criteria could be considered.

A theoretical motivation for objective function (10) can be obtained from Theorem 1 of [7], which bounds the generalization error of DDAG (Decision Directed Acyclic Graph) classifiers. The bound depends of the radius R of the smallest ball containing the training data. To use this result, let us note that decision criterion (7) can be implemented using pairwise boolean classifiers $f_{ij}(\mathbf{x})(\mathbf{x}) = I\left((\mathbf{v}_i - \mathbf{c})^T(\mathbf{x} - \mathbf{c}) > (\mathbf{v}_j - \mathbf{c})^T(\mathbf{x} - \mathbf{c})\right)$ organized in a DDAG. Note now that the decision function only depends on the training data involved in the definition of the prototypes $\mathbf{v}_k$. Other points could be removed from the training set, leaving the same classifier. Choosing the prototypes as close as possible works hence by reducing the radius of the ball containing the minimal equivalent training set.

An algorithm employing criterion (10) to build a multicategory classifier is presented in [3]. In this case, however, the prototypes are not confined to the convex hull corresponding to the class they represent and the decision function is defined as in (7), without considering a "convenient" reference point $\mathbf{c}$. Moreover this algorithm explicitly imposes constraints to deal with classification error and has hence $m \cdot K$ constraints which become variables in the dual optimization problem.

4 The Algorithm and its Extensions

Since each prototype vector $\mathbf{v}_k$ is a point in the convex hull H_k, it can be expanded in terms of the examples $\mathbf{X}_k$ as $\mathbf{v}_k = \mathbf{X}_k^T \mathbf{u}_k$, where the vector of parameters $\mathbf{u}_k$ satisfies the convexity constraints. The optimization problem (10) hence becomes

$$\begin{aligned} \text{minimize}_{\{\mathbf{u}_i\}} \quad & \frac{1}{2} \sum_l \sum_{m<l} \|\mathbf{X}_l^T \mathbf{u}_l - \mathbf{X}_m^T \mathbf{u}_m\|^2 \\ \text{s.t.} \quad & \mathbf{1}^T \mathbf{u}_i = 1, \;\; \mathbf{u}_i \geq 0 \;\; i = 1, 2, \ldots, K \end{aligned} \tag{14}$$

Calculating on the objective function

$$\begin{aligned} & \|\mathbf{X}_l^T \mathbf{u}_l - \mathbf{X}_m^T \mathbf{u}_m\|^2 \\ &= (\mathbf{X}_l^T \mathbf{u}_l - \mathbf{X}_m^T \mathbf{u}_m)^T (\mathbf{X}_l^T \mathbf{u}_l - \mathbf{X}_m^T \mathbf{u}_m) \\ &= \mathbf{u}_l' \mathbf{K}^{ll} \mathbf{u}_l - 2\mathbf{u}_l' \mathbf{K}_{lm} \mathbf{u}_m + \mathbf{u}_m' \mathbf{K}_{mm} \mathbf{u}_m \end{aligned} \tag{15}$$

where $\mathbf{K^{ij}} = \mathbf{X}_i \mathbf{X}_j^T$ is the matrix of dot products among the examples of class i and class j. We have then that the objective function is

$$\begin{aligned} & \sum_l \sum_{m \neq l} \|\mathbf{X}_l^T \mathbf{u}_l - \mathbf{X}_m^T \mathbf{u}_m\|^2 \\ &= (K-1) \sum_l \mathbf{u}_l' \mathbf{K}^{ll} \mathbf{u}_l - 2 \sum_l \sum_{m<l} \mathbf{u}_l' \mathbf{K}_{lm} \mathbf{u}_m \end{aligned} \tag{16}$$

Now, we can write the products $\mathbf{u}_i^T \mathbf{K^{ij}} \mathbf{u}_j$ component by component to obtain

$$\begin{aligned} & (K-1) \sum_l \sum_{i \in I_l} \sum_{j \in I_l} u_{l,i} \left(\mathbf{x}_{l,i}^T \mathbf{x}_{l,i}\right) u_{l,i} \\ & - \sum_l \sum_{m \neq l} \sum_{i \in I_l} \sum_{j \in I_m} u_{l,i} \left(\mathbf{x}_{l,i}^T \mathbf{x}_{m,j}\right) u_{m,j} \end{aligned} \tag{17}$$

where $u_{s,r}$ is the r-th component of $\mathbf{u}_s$ and $\mathbf{x}_{s,r}$ is the r-th example of the class s. Note that in the last expression all the different dot products among training examples $\mathbf{x}_i^T \mathbf{x}_j$ appear exactly once, with an scalar α_{ij} which depends of the classes of x_i and x_j. If the classes coincide ($y_i = y_j$) we have $\alpha_{ij} = (K-1)$ and if the classes do not match ($y_i \neq y_j$) $\alpha_{ij} = -1$. We can hence write the objective function (17) more compactly by taking all the dot products among training examples, and using just one parameter vector $\mathbf{u} = (\mathbf{u}_1^T \; \mathbf{u}_2^T \; \ldots, \mathbf{u}_K^T)^T \in \mathbb{R}^m$ composed of all the class-dependent parameter vectors $\mathbf{u}_i$,

$$\sum_{i \in I} \sum_{j \in I} u_i \left(\alpha_{ij} \mathbf{x}_i^T \mathbf{x}_j\right) u_j \tag{18}$$

The latter expression actually corresponds to the (pseudo-)norm of $\mathbf{u}$ induced by the matrix $\bar{\mathbf{K}} = (\bar{k}_{ij})$ where $\bar{k}_{ij} = \alpha_{ij}(\mathbf{x}_i^T \mathbf{x}_j)$. Our optimization problem takes finally the same form as the binary SVM:

$$\begin{aligned} & \texttt{AD-SVM:} \\ \text{minimize}_{\{\mathbf{u}\}} \quad & \|\mathbf{u}\|_{\bar{\mathbf{K}}} = \mathbf{u}^T \bar{\mathbf{K}} \mathbf{u} \\ \text{s.t.} \quad & \sum_{i \in I_j} u_i = 1 \; \forall j, \;\; 0 \leq u_i \leq 1 \; \forall i \end{aligned} \tag{19}$$

In order to show that $\bar{\mathbf{K}}$ is positive semidefinite, note that $\bar{\mathbf{K}}$ corresponds to the Hadamard product $\mathbf{K} \circ \mathbf{A}$, where $\mathbf{K} = (k_{ij})$ is the gram matrix induced by the training patterns $k_{ij} = \mathbf{x}_i^T \mathbf{x}_j$ and $\mathbf{A} = (\alpha_{ij})$ is the gram matrix induced by a set of vector-valued codes associated to these patterns. To

encode the class associated with a pattern $\mathbf{x}_i$, we define the K-dimensional vector $\mathbf{t}_i$ with $K - 1/\sqrt{K}$ in the coordinate corresponding to the class of $\mathbf{x}_i$ and $-1/\sqrt{K}$ elsewhere, that is $t_{i,y_i} = K - 1/\sqrt{K}$ and $t_{i,j} = -1/\sqrt{K}\ \forall j \neq y_i$. We then have that $\alpha_{ij} = \mathbf{t}_i^T \mathbf{t}_j$ which shows that not only $\mathbf{K}$ but also $\mathbf{A}$ is a gram matrix. From here, it follows that $\bar{\mathbf{K}} = \mathbf{K} \circ \mathbf{A}$ is positive semidefinite. This also probes that $\mathbf{u}^T \bar{\mathbf{K}} \mathbf{u}$ in (19) is actually a pseudo-norm. Interestingly, the vector-valued codes $\mathbf{t}_i$ are equivalent, up to a scaling by $\sqrt{K}/K - 1$, to the codes employed in [6] to implement multicategory classifiers.

In order to build non-linear extensions of the classifier by using kernels, not only the optimization function but also the decision function (7) should be written in terms of dot products among training patterns. Let us note that since the component $\mathbf{c}^T(\mathbf{x} - \mathbf{c})$ of this function is constant among the different classes, the decision criteria can be reduced to

$$\hat{f}(\mathbf{x}) \quad = \quad \arg\max_i \mathbf{v}_i^T (\mathbf{x} - \mathbf{c}) \tag{20}$$

Now, we can write each prototype $\mathbf{v}_i$ in terms of the examples of its class, $\mathbf{v}_k = \mathbf{X}_k^T \mathbf{u}_k$. On the other hand, $\mathbf{c} = 1/K \sum_i \mathbf{v}_i$ and hence we have

$$\begin{aligned}
\mathbf{v}_i^T(\mathbf{x} - \mathbf{c}) &= \mathbf{v}_i^T \mathbf{x} - \mathbf{v}_i^T \mathbf{x}\mathbf{c} \qquad (21)\\
&= \mathbf{u}_k^T \mathbf{X}_k \mathbf{x} - \frac{1}{K} \sum_i \mathbf{u}_k^T \mathbf{X}_k \mathbf{X}_i^T \mathbf{u}_k \\
&= \sum_{k \in I_i} u_k \mathbf{x}_k^T \mathbf{x} - \frac{1}{K} \sum_{k \in I_i} \sum_j \sum_{l \in I_j} u_k \mathbf{x}_k^T \mathbf{x}_l u_l \\
&= \sum_{k \in I_i} u_k \mathbf{x}_k^T \mathbf{x} - \frac{1}{K} \sum_{k \in I_i} \sum_{l \in I} u_k \mathbf{x}_k^T \mathbf{x}_l u_l
\end{aligned}$$

which only depends on dot products among training examples. Now we consider two extensions of the proposed classifier. The first is by introducing the idea of reduced convex hulls. Reduced convex hulls are used in binary SVMs to relax the linear separability assumption and are built by bounding by a constant $0 < \eta < 1$ the influence that each example can have in the expansion defining the points $v_1 \in H_1$ and $v_2 \in H_2$. This has the effect of contracting the original convex hulls. Parameter η has hence the same role as in the standard soft margin version of binary SVMs. Considering the reduced versions of the convex hulls enclosing each class, our classifier has the ability to reduce the influence of some outlying observations. The soft-margin version of our classifier is hence,

Soft AD-SVM:

$$\begin{aligned}
\text{minimize}_{\{\mathbf{u}\}} \quad & \|\mathbf{u}\|_{\bar{\mathbf{K}}} = \mathbf{u}^T \bar{\mathbf{K}} \mathbf{u} \qquad (22)\\
\text{s.t.} \quad & \sum_{i \in I_j} u_i = 1\ \forall j,\ \ 0 \leq u_i \leq \eta\ \forall i
\end{aligned}$$

Algorithm 1 AD-SVM (Extended Version)

1: Let $S = \{(\mathbf{x}_i, y_i); i = 1, \ldots, m\}$ be the set of examples where $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_K\}$. Let I be the complete set of indexes $\{1, 2, \ldots, m\}$ and $I_i \subset I$ the set of indices corresponding to examples of class i.

2: Build the kernel matrix of dot products $\mathbf{K} = (k_{ij})$ $i, j \in I$ as $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, where k is a Mercer kernel

3: Define $\bar{\mathbf{K}} = (\bar{k}_{ij})$, with $\bar{k}_{ij} = (K-1)k_{ij}$ if $y_i = y_j$ and $\bar{k}_{ij} = -k_{ij}$ otherwise.

4: Solve

$$\begin{aligned}
\text{minimize}_{\{\mathbf{u}\}} \quad & \mathbf{u}^T \bar{\mathbf{K}} \mathbf{u} = \sum_{i,j \in I} u_i \bar{k}_{ij} u_j \qquad (23)\\
\text{s.t.} \quad & \sum_{i \in I_j} u_i = 1\ \forall j,\ \ 0 \leq u_i \leq \eta\ \forall i
\end{aligned}$$

where parameter η defines the degree of convex hull reduction.

5: Build the decision function as

$$\begin{aligned}
\hat{f}(\mathbf{x}) &= \arg\max_j \sum_{i \in I_j} u_i k(\mathbf{x}_i, \mathbf{x}) - b_j \qquad (24)\\
b_j &= \frac{1}{K} \sum_{i \in I_j} \sum_{k \in I} u_i k(\mathbf{x}_i, \mathbf{x}_k) u_k \qquad (25)
\end{aligned}$$

6: Classify new instances $\mathbf{x}$ as $\mathcal{C}_j$ if $f(\mathbf{x}) = j$.

The final extension consists in building a non-linear classifier by using Mercer kernels. This is straightforward since our objective and decision function both depend explicitly on dot products which can be replaced by more general kernels $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ implicitly mapping $\mathcal{X}$ to a higher dimensional space $\Phi(\mathcal{X})$ in which we build our linear classifier. Algorithm (1) summarizes our method for multicategory classification considering the two extensions of the basic linear classifier. It should be noted that this also preserves the form of the binary SVM.

5 Experiments and Conclusions

We present experimental results in several UCI classification benchmarks [2], comparing the proposed technique (called *ad-svm*) with two widely used extensions of support vector machines for multicategory classification: the one-versus-the-rest (called *ova*) [8] and the one-versus-one (called *ovo*) [5] schemes, already compared with other single objective extensions for example in [4]. The experimental setup is as follows. A randomly selected 20% of the examples of each dataset were reserved to build the testing set. On the resting 80%, 10-fold cross-validation was used for parameter selection. Performance is finally evaluated as the

Dataset	Alg.	TA		CV-T	
		hard	*soft*	*hard*	*soft*
Iris	*ad-svm*	0.950	0.950	0.66	0.65
	ova	0.933	0.933	1.74	1.74
	ovo	0.933	0.933	0.55	0.55
Wine	*ad-svm*	0.994	0.994	0.60	0.59
	ova	0.994	0.994	25.63	25.62
	ovo	0.994	0.994	0.93	0.95
Soybean	*ad-svm*	1.000	1.000	0.06	0.06
	ova	1.000	1.000	0.31	0.32
	ovo	1.000	1.000	0.43	0.44
Waveform	*ad-svm*	0.838	0.843	24.63	31.90
	ova	0.818	0.843	40.07	62.07
	ovo	0.815	0.843	15.15	21.59
Ecoli	*ad-svm*	0.737	0.793	3.56	1.25
	ova	0.763	0.848	54.24	16.28
	ovo	0.785	0.844	9.69	9.26

Table 1. Performance of the different algorithms: TA corresponds to the testing accuracy and CV-T to the time in seconds incurred in the cross-validation process, that is it corresponds to 10 training rounds.

fraction of the testing examples correctly classified. Before training, each data set was normalized to have zero mean and unitary covariance matrix. Table (1) shows the results for the algorithms *ad-svm*, *ova* and *ovo*. For all the algorithms, a RBF kernel $k(x, y) = exp(-\|x - y\|^2/\sigma^2)$ was employed. To determine the value of σ, a grid search between 10^{-4} and 10^{+4} was performed. Column *hard* corresponds to the results obtained without convex hull reduction ($\eta = 1$) both for the *ad-svm* objective function and the binary SVMs implemented by the decomposition approaches. Column *soft* corresponds to the results obtained with convex hull reduction ($\eta \leq 1$). Convex hull relaxation parameter η was determined using the kernel bandwidth parameter found in the experiments without convex-hull reduction. A grid search between 10^{-3} and 10^{0} was carried out.

From table (1) we can see that the proposed algorithm is considerably faster than the one-versus-the-rest approach. This result is in fact predictable from the definition of the classifier: this solves just one problem with m variables and $2m + K$ constraints while one-versus-the-rest solves K problems of m variables and m constraints. With respect to the the one-versus-one scheme, the pattern is not so clear: ad-svm is faster three times. Since one-versus-one solves a set of problems of smaller size, dealing with two classes each time, the advantage or disadvantage of the method depends on the complexity of the solver. Supposing that the problem is class-balanced, and the solver has time-complexity $O(m^d)$ in the number of examples m, we would have that one-versus-one has a complexity proportional to $(\frac{m}{K})^d \cdot K^2 = K^{2-d}m^2$, while the complexity of our method is around m^2. If $d > 2$, one-versus-one is superior. When $d = 2$, time complexity is the same. When $d < 2$, in contrast, our method begins to obtain advantages. For our experiments we used the *quadprog* solver of MatLab. It is clear however that an SMO-like training algorithm or other decomposition approach would allow us to obtain greater and more systematic improvements. Research in this direction is being carried out.

In terms of prediction accuracy we can see that the algorithms obtain comparable results, with two wins, two ties and one loss for our algorithm in the experiments without convex-hull reduction. Allowing $\eta < 1$ we can observe systematic improvements for all the algorithms.

As a final remark, we would like to note that in contrast to *ad-svm*, decomposition approaches like *ova* an *ovo* are combining independently trained binary SVMs and, hence, they are actually selecting more than one prototype to model boundaries between classes. One-versus-one, for example, selects different prototypes for each one of the $\binom{n}{2}$ pairwise boundaries. Experiments in this work shows that similar prediction error can be obtained by using just one convex-hull prototype selected and combined appropriately in a decision function which has the binary SVM as a special case. New research is being carried out to refine this base procedure.

References

[1] K. P. Bennett and E. J. Bredensteiner. Duality and geometry in svm classifiers. In *Proceedings of the Seventeenth ICML*, pages 57–64, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[2] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.

[3] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, (2):265–292, 2001.

[4] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

[5] U. Kressel. Pairwise classification and support vector machines. In *Advances in kernel methods: support vector learning*, pages 255–268. MIT Press, Cambridge, MA, USA, 1999.

[6] Y. Lee, Y. Li, and G. Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

[7] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. *Proceedings of NIPS'99*, pages 547–553, 1999.

[8] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.