Adapting Two-Class Support Vector Classification Methods to Many Class Problems

Simon I. Hill SIH22@ENG.CAM.AC.UK
Department of Engineering, University of Cambridge, UK

Arnaud Doucet ARNAUD@CS.UBC.CA
Departments of Computer Science and Statistics, University of British Columbia, Canada

Abstract

A geometric construction is presented which is shown to be an effective tool for understanding and implementing multi-category support vector classification. It is demonstrated how this construction can be used to extend many other existing two-class kernel-based classification methodologies in a straightforward way while still preserving attractive properties of individual algorithms. Reducing training times through incorporating the results of pairwise classification is also discussed and experimental results presented.

1. Introduction

Recently (Vapnik, 1998; Van Gestel et al., 2001; Mangasarian & Musicant, 2001; Fung & Mangasarian, 2001b; Herbrich et al., 2000, for example) there have been a large number of kernel-based, two-class classification algorithms developed. Often though problems require classification into $M > 2$ classes and it becomes important to understand how to extend a particular methodology.

Three dominant approaches exist for achieving this, of which two are *one-against-all* and *pairwise* or *one-against-one* methods. Briefly, the one-against-all technique sets up a series of M two-class problems each discriminating one of the M classes against the rest. By contrast the pairwise method uses $\frac{M(M-1)}{2}$ different classifiers, each concentrating on the comparison between two individual categories.

Appearing in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005. Copyright 2005 by the author(s)/owner(s).

These two techniques have been shown by Allwein et al. (2001) to be special cases of the third main approach; *Error Correcting Output Codes (ECOCs)* (Dietterich & Bakiri, 1995). ECOCs typically involve the construction of some binary code word[1], the determination of each bit of which involves an individual classification. For instance, each of eight classes can be individually assigned unique binary code words of length three. Unfortunately, as in this example, ECOCs do not always have equivalent Hamming distances between classes, which suggests that not all classes are being compared against each other equivalently.

One-against-all and pairwise methods are exceptions to this, however even with these the end results must be in some way combined to produce a single classification of a new input. Clearly, some heuristic, not directly related to the initial training, must be used to provide the final decision, be it through binary codewords, a voting method, or some other approach.

This use of heuristics may well be adequate, contributions such as that by Rifkin and Klautau (2004) argue that one-against-all and pairwise methods can be made to perform practically as well as other methods. This should come as no surprise, when comparing the performance of algorithms applied to a variety of datasets their individual implicit models will be better suited to some, and less well suited to others.

Often however it is desirable to have a clear understanding of the optimisation process and a significant contribution of the framework used in this work is that within it many single optimisation methods can be understood and directly compared. Indeed, many recent Support Vector (SV) contributions have sought

[1] An exception to this being the case of pairwise comparison when code word elements take values from $\{-1, 0, 1\}$.

to do this by using what are known as *all-together* approaches (Hsu & Lin, 2002a).

The very fact that so many different efforts have been made to find a method involving a single optimisation which is competitive in terms of speed is in itself evidence of a desire by the research community to overcome heuristic solutions. Unfortunately though, all-together methods often have the serious drawback that the optimisation process can be extremely computationally intensive, typically orders of magnitude more so than the pairwise approach, for instance.

In this paper we introduce a geometric structure which aids considerably in understanding connections between different multi-category classification methods. Additionally it enables more efficient algorithms to be developed which can take advantage of the insight provided to streamline the optimisation process. It becomes possible, for example, to perform pairwise optimisation, map the results obtained into this geometric structure, and fine-tune the result to finally obtain the all-together endpoint.

A brief introduction to the general approach employed is given in Section 2. Section 3 then aims to give an overview of how this construction is applicable to a wide range of similar algorithms. Implementation issues are briefly discussed in Section 4.

2. The Geometric Construction

We consider the situation in which the aim is to find a classifier which will take some input $\mathbf{x} \in \mathcal{X}$ and output a corresponding class $\vartheta \in \Theta$, of which there are M. This classifier is to be found based on training data $\{(\mathbf{x}_i, \vartheta_i) \in \mathcal{X} \times \Theta\}_{i=1}^{N}$.

Class determination of some input from the set $\mathcal{X}$ is performed in the binary classification case by considering the sign of an underlying real-valued function $f : \mathcal{X} \to \mathbb{R}$ (Vapnik, 1998, for example). In considering the M-class case, the underlying vector-valued function $\mathbf{f} : \mathcal{X} \to \mathbb{R}^{M-1}$ will be found, where $\mathbf{f} = \left[\begin{array}{ccc} f_1 & \dots & f_{M-1} \end{array}\right]^T$. The idea behind the use of an $(M-1)-$dimensional space is to be able to introduce M equally separable class-target vectors, $\{\mathbf{y}(\vartheta) : \vartheta \in \Theta\}$. The class of input $\mathbf{x}$ will be determined by identifying to which class-target vector the output $\mathbf{f}(\mathbf{x})$ is closest.

Many binary classification algorithms actually implement such a decision, where classes, denoted A and B, have class targets $y(A) = -1$ and $y(B) = +1$. Consider now that a third class, C, is a possibility. Clearly a one-dimensional numerical label is insufficient for the classes to be equidistant[2]. A two-dimensional arrangement as illustrated in Figure 1 does, however, allow this. Here the class-target vectors are

$$\begin{aligned} \mathbf{y}(A) &= \left[\begin{array}{cc} -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{array}\right]^T, \\ \mathbf{y}(B) &= \left[\begin{array}{cc} \frac{\sqrt{3}}{2} & -\frac{1}{2} \end{array}\right]^T, \\ \mathbf{y}(C) &= \left[\begin{array}{cc} 0 & 1 \end{array}\right]^T. \end{aligned} \tag{1}$$

where $\|\mathbf{y}(\vartheta)\| = 1$ for all classes ϑ (note that in this work $\|\cdot\|$ denotes the 2-norm of a vector, *i.e.* $\|\mathbf{y}\| = \sqrt{y_1^2 + \cdots + y_{M-1}^2}$ and, furthermore, normalisation will imply $\frac{\mathbf{y}}{\|\mathbf{y}\|}$) as this improves tractability later.

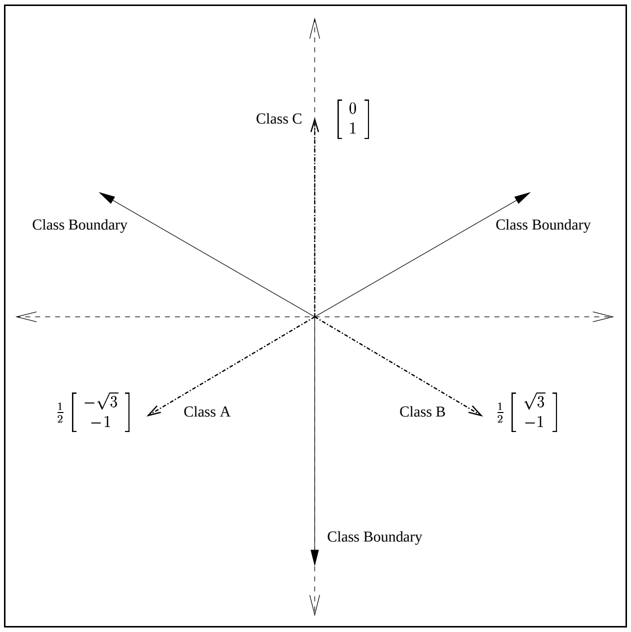
Class C
$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$
Class Boundary
Class Boundary
$\frac{1}{2}\begin{bmatrix} -\sqrt{3} \\ -1 \end{bmatrix}$
Class A
Class B
$\frac{1}{2}\begin{bmatrix} \sqrt{3} \\ -1 \end{bmatrix}$
Class Boundary

Figure 1. Diagram showing class labels for classification into three. *The class-target vectors corresponding to classes A, B and C are shown. The class boundaries are given by solid lines.*

These are example class-target vectors, however, in general it is important to understand that the optimisation methods which will be described are applicable regardless of their rotation. Indeed, although the apparent Cartesian coordinate asymmetry may seem unintuitive, the important consideration is the relative positioning of class-target vectors with respect to each other. The optimisation procedure has no dependence on any particular orientation. This will be discussed a little more in Subsection 3.1.

[2]Note that in the case that little is known about the relationship between the classes then the logical arrangement is for them to be equidistant.

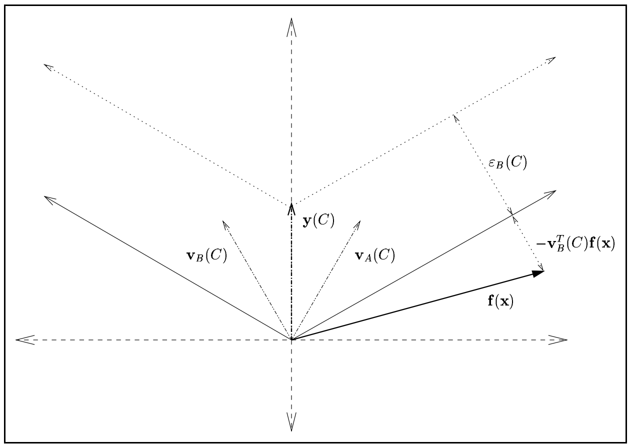

Figure 2. Diagram showing elements involved in determining empirical loss associated with a training sample of class C.

The same approach to that described for $M = 3$ is taken when considering larger values of M. While typically $M = 3$ will be used in this work as an example case, extensions to higher values of M follow without too much further consideration.

In setting up the classification process, each class is assigned a subspace of the $(M-1)$ dimensional output space. In particular these subspaces are the Voronoi regions associated with the class targets. As a result, class boundaries can be found by forming hyperplanes between class regions which consist of all points equidistant from the two relevant class targets. Observing in which of the regions $\mathbf{f}(\mathbf{x})$ lies gives $h(\mathbf{x})$,

$$h(\mathbf{x}) = \text{The class of the region in which } \mathbf{f}(\mathbf{x}) \text{ lies.} \tag{2}$$

In order to describe the empirical losses employed in the various algorithms, it is first necessary to define the vectors perpendicular to the hyperplane dividing the region between $\mathbf{y}(A)$ and $\mathbf{y}(B)$. Let

$$\mathbf{v}_A(B) = \frac{\mathbf{y}(B) - \mathbf{y}(A)}{\|\mathbf{y}(B) - \mathbf{y}(A)\|}. \tag{3}$$

These vectors are illustrated for class C in Figure 2 in which a margin $\varepsilon_B(C)$ is also introduced. This margin is often used in defining the empirical loss. The fundamental starting point in determining and combining individual loss components for many algorithms is also illustrated in Figure 2. Here a training point $\mathbf{x}$ with class C has $\mathbf{f}(\mathbf{x})$ which falls outside the required region. This is penalised in multi-category Support Vector Classification (SVC) by $\big(\varepsilon_B(C) - \mathbf{v}_B^T(C)\mathbf{f}(\mathbf{x})\big)$ in an analogous way to the binary SVC empirical loss of $(1 - yf(\mathbf{x}))$. Indeed in the binary case $(v_B(A), v_A(B)) = (y(A), y(B))$ and $\varepsilon_A(B) = \varepsilon_B(A) = 1$.

3. Algorithms and their extensions

We restrict our attention in this paper to SVC §3.1, $\nu-$Support Vector Classification ($\nu-$SVC) §3.2, Least Squares Support Vector Classification (LS-SVC) §3.3, Lagrangian Support Vector Classification (LSVC) §3.4, Proximal Support Vector Classification (PSVC) §3.5, and Bayes Point Machines (BPMs) §3.6. Multi-category extensions of many of these have been presented, indeed there are many SVC methodologies alone (Lee et al., 2004; Hsu & Lin, 2002a; Crammer & Singer, 2001; Weston & Watkins, 1999; Kreßel, 1999; Vapnik, 1998, for example) however with some manipulation these can be seen to be special cases of the methodology outlined here.

3.1. Standard Support Vector Classification

Multi-category SVC is presented here as a starting point. Recall (Vapnik, 1998) that the two-class SVC optimisation problem is to

$$\begin{aligned} &\text{minimise} \ \left(\frac{1}{2}\|\mathbf{w}\|_{\mathcal{F}}^2 + C\sum_{i=1}^{N}\xi_i\right) \\ &\text{subject to} \ \begin{cases} y_i\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}\rangle_{\mathcal{F}} + b\right] \geq 1 - \xi_i \\ \xi_i \geq 0. \end{cases} \end{aligned} \tag{4}$$

The multi-category extension to this is to

$$\begin{aligned} &\min \ \left(\frac{1}{2}\sum_{m=1}^{M-1}\|\mathbf{w}_m\|_{\mathcal{F}}^2 + C\sum_{i=1}^{N}\sum_{\psi\in(\Theta-\vartheta_i)}\xi_{i,\psi}\right) \\ &\text{sub. to} \ \begin{cases} \sum_{m=1}^{M-1} v_{\psi,m}(\vartheta_i)\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}_m\rangle_{\mathcal{F}} + b_m\right] \\ \quad \geq \varepsilon_\psi(\vartheta_i) - \xi_{i,\psi} \\ \xi_{i,\psi} \geq 0 \end{cases} \end{aligned} \tag{5}$$

where $v_{\psi,m}(\vartheta)$ is the mth element of $\mathbf{v}_\psi(\vartheta)$. The familiar Lagrangian dual approach can be used to find an optimal solution to this. Let

$$\mathbf{V}(\vartheta) = \left[\ \mathbf{v}_A(\vartheta) \quad \mathbf{v}_B(\vartheta) \quad \ldots \quad \mathbf{v}_{\psi\neq\vartheta}(\vartheta) \quad \ldots\ \right] \tag{6}$$

and represent the mth row of $\mathbf{V}(\vartheta)$ by $\mathbf{v}_m^{*T}(\vartheta)$. With this then the dual becomes

$$\begin{aligned} L_D = &-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{m=1}^{M-1}\boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j K(\mathbf{x}_i, \mathbf{x}_j) \\ &+\sum_{i=1}^{N}\boldsymbol{\alpha}_i^T\boldsymbol{\varepsilon}(\vartheta_i) \end{aligned} \tag{7}$$

where,

$$\begin{aligned}\boldsymbol{\alpha}_i &= \left[\begin{array}{ccccc} \alpha_{i,A} & \alpha_{i,B} & \dots & \alpha_{i,\psi\neq\vartheta_i} & \dots \end{array}\right]^T \\ \boldsymbol{\varepsilon}(\vartheta_i) &= \left[\begin{array}{ccccc} \varepsilon_A(\vartheta_i) & \varepsilon_B(\vartheta_i) & \dots & \varepsilon_{\psi\neq\vartheta_i}(\vartheta_i) & \dots \end{array}\right]^T\end{aligned}$$

and the kernel function has been denoted $K(\mathbf{x}_i, \mathbf{x}_j) = \langle\Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle_{\mathcal{F}}$. The derivation of equation (7) also introduces the constraints that $C \geq \alpha_{i,\psi} \geq 0$, $\forall i, \psi \in (\Theta - \vartheta_i)$ and $\sum_{i=1}^N \mathbf{V}(\vartheta_i)\boldsymbol{\alpha}_i = \mathbf{0}$. Finally, note that having found $\mathbf{b}$, the function $\mathbf{f}(\cdot)$ can be expressed,

$$\mathbf{f}(\mathbf{x}) = \sum_{i=1}^N \mathbf{V}(\vartheta_i)\boldsymbol{\alpha}_i K(\mathbf{x}, \mathbf{x}_i) + \mathbf{b}. \tag{8}$$

A final issue to consider is that of rotational invariance to the structuring of the problem — as initially raised in Section 2. Note that the only influence of rotational orientation in equation (7) is through the summation term $\boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j K(\mathbf{x}_i, \mathbf{x}_j)$. Consider now that the chosen orientation is rotated in some way as described by a rotation matrix $\mathbf{R}$, this quadratic term then becomes,

$$\begin{aligned}&\boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{R}^T\mathbf{R}\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j K(\mathbf{x}_i, \mathbf{x}_j) \\ &\qquad = \boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j K(\mathbf{x}_i, \mathbf{x}_j)\end{aligned} \tag{9}$$

due to the fact that rotation matrices are orthonormal. Hence the optimisation problem is rotationally invariant. A similar argument can also be used for the other algorithms considered here.

3.2. $\nu-$Support Vector Classification

In this case the two-class optimisation problem (Schölkopf & Smola, 2002) is given by

$$\begin{aligned}&\text{minimise } \left(\frac{1}{2}\|\mathbf{w}\|_{\mathcal{F}}^2 + \sum_{i=1}^N \xi_i - \nu\varepsilon\right) \\ &\text{subject to } \left\{\begin{array}{l} y_i\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}\rangle_{\mathcal{F}} + b\right] \geq \varepsilon - \xi_i \\ \xi_i \geq 0, \text{ and, } \varepsilon \geq 0 \end{array}\right.\end{aligned} \tag{10}$$

and the extension to the *polychotomous* case is straightforward,

$$\begin{aligned}&\text{minimise } \left(\frac{1}{2}\sum_{m=1}^{M-1}\|\mathbf{w}_m\|_{\mathcal{F}}^2\right. \\ &\qquad\left. + \sum_{i=1}^N \sum_{\psi\in(\Theta-\vartheta_i)} \xi_{i,\psi} - \sum_{\vartheta\in\Theta}\sum_{\psi\in(\Theta-\vartheta)} \nu\varepsilon_\psi(\vartheta)\right) \\ &\text{subject to } \left\{\begin{array}{l} \sum_{m=1}^{M-1} v_{\psi,m}(\vartheta_i)\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}_m\rangle_{\mathcal{F}} + b_m\right] \\ \quad \geq \varepsilon - \xi_{i,\psi} \\ \xi_{i,\psi} \geq 0, \text{ and, } \varepsilon \geq 0. \end{array}\right.\end{aligned} \tag{11}$$

Following the usual Lagrangian dual approach (Schölkopf & Smola, 2002) results in the final aim being to maximise

$$L_D = -\frac{1}{2}\sum_{i=1}^N\sum_{j=1}^N\sum_{m=1}^{M-1} \boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{12}$$

subject to $0 \leq \alpha_{i,\phi} \leq 1$, $\forall i, \psi \in (\Theta - \vartheta_i)$, $\sum_{i=1}^N \mathbf{V}(\vartheta_i)\boldsymbol{\alpha}_i = \mathbf{0}$, and $\sum_{i=1}^N \sum_{\psi\in(\Theta-\vartheta_i)} \alpha_{i,\psi} > \nu$. The output is as given in equation (8).

3.3. Least Squares Support Vector Classification

LS-SVC as developed at length by Van Gestel et al. (2001) is similar to standard SVC, however now with quadratic instead of linear loss, and inequality constraints replaced by equality ones. Multiclass versions have been published (Van Gestel et al., 2002) which rely on coding schemes, however these can lead to inconsistent treatment of different classes. The two-class case aims to

$$\begin{aligned}&\text{minimise } \left(\frac{1}{2}\|\mathbf{w}\|_{\mathcal{F}}^2 + C\sum_{i=1}^N \xi_i^2\right) \\ &\text{subject to } y_i\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}\rangle_{\mathcal{F}} + b\right] = 1 - \xi_i\end{aligned} \tag{13}$$

A multi-category extension follows succinctly,

$$\begin{aligned}&\text{minimise } \left(\frac{1}{2}\sum_{m=1}^{M-1}\|\mathbf{w}_m\|_{\mathcal{F}}^2 + C\sum_{i=1}^N\sum_{\psi\in(\Theta-\vartheta_i)} \xi_{i,\psi}^2\right) \\ &\text{subject to } \sum_{m=1}^{M-1} v_{\psi,m}(\vartheta_i)\left[\langle\Phi(\mathbf{x}_i), \mathbf{w}_m\rangle_{\mathcal{F}} + b_m\right] \\ &\qquad = \varepsilon_\psi(\vartheta_i) - \xi_{i,\psi}.\end{aligned} \tag{14}$$

Now, define $\boldsymbol{\alpha}' = \left[\begin{array}{ccc} \boldsymbol{\alpha}_1^T & \dots & \boldsymbol{\alpha}_N^T \end{array}\right]^T$, $\boldsymbol{\varepsilon}' = \left[\begin{array}{ccc} \boldsymbol{\varepsilon}^T(\vartheta_1) & \dots & \boldsymbol{\varepsilon}^T(\vartheta_N) \end{array}\right]^T$, $\mathbf{Z}_m = \left[\begin{array}{ccc} \Phi(\mathbf{x}_1)\mathbf{v}_m^{*T}(\vartheta_1) & \dots & \Phi(\mathbf{x}_N)\mathbf{v}_m^{*T}(\vartheta_N) \end{array}\right]$, $\mathbf{V}' = \left[\begin{array}{ccc} \mathbf{V}(\vartheta_1) & \dots & \mathbf{V}(\vartheta_N) \end{array}\right]$, and $\mathbf{Z}' = \left[\begin{array}{ccc} \mathbf{Z}_1^T & \dots & \mathbf{Z}_{M-1}^T \end{array}\right]^T$. With these definitions then it can be shown (Van Gestel et al., 2001) the optimisation problem becomes equivalent to finding $\boldsymbol{\alpha}'$ and $\mathbf{b}$ to satisfy,

$$\left[\begin{array}{cc} \mathbf{0} & \mathbf{V}' \\ \mathbf{V}'^T & \mathbf{Z}'^T\mathbf{Z}' + C\mathbf{I} \end{array}\right]\left[\begin{array}{c} \mathbf{b} \\ \boldsymbol{\alpha}' \end{array}\right] = \left[\begin{array}{c} \mathbf{0} \\ \boldsymbol{\varepsilon}' \end{array}\right]. \tag{15}$$

The classifier is found by solving these linear equations. Note that finding $\mathbf{Z}'^T\mathbf{Z}'$ does not require reference to the feature space, but only kernel evaluations. The final output is again as in equation (8).

3.4. Lagrangian Support Vector Classification

As introduced by Mangasarian and Musicant (2001), the LSVC is an algorithm which has its strength in that it is computationally efficient, and easy to implement. The method for two-class classification aims to

$$\text{minimise} \quad \left(\frac{1}{2}\left[\|\mathbf{w}\|_{\mathcal{F}}^2 + b^2\right] + C\sum_{i=1}^{N}\xi_i^2\right) \qquad (16)$$
$$\text{subject to } y_i\left[\langle\Phi(\mathbf{x}_i),\mathbf{w}\rangle_{\mathcal{F}} + b\right] \geq 1 - \xi_i.$$

This can be reformulated to a multi-category problem resulting in,

$$\text{minimise} \quad \left(\frac{1}{2}\sum_{m=1}^{M-1}\left[\|\mathbf{w}_m\|_{\mathcal{F}}^2 + b_m^2\right] + C\sum_{i=1}^{N}\sum_{\psi\in(\Theta-\vartheta_i)}\xi_{i,\psi}^2\right)$$
$$\text{subject to } \sum_{m=1}^{M-1} v_{\psi,m}(\vartheta_i)\left[\langle\Phi(\mathbf{x}_i),\mathbf{w}_m\rangle_{\mathcal{F}} + b_m\right] \geq \varepsilon_\psi(\vartheta_i) - \xi_{i,\psi}. \qquad (17)$$

The dual to this is,

$$L_D = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{m=1}^{M-1}\boldsymbol{\alpha}_i^T\mathbf{V}^T(\vartheta_i)\mathbf{V}(\vartheta_j)\boldsymbol{\alpha}_j\left[K(\mathbf{x}_i,\mathbf{x}_j)+1\right] + \sum_{i=1}^{N}\boldsymbol{\alpha}_i^T\left[\boldsymbol{\varepsilon}(\vartheta_i) - \frac{1}{2C}\boldsymbol{\alpha}_i\right] \qquad (18)$$

which needs to be maximised subject to $\alpha_{i,\psi} \geq 0$ for all i and all $\psi \in (\Theta - \vartheta_i)$, and once that has been done then $\mathbf{b} = \sum_{i=1}^{N}\mathbf{V}(\vartheta_i)\boldsymbol{\alpha}_i$. The final solution again takes the form of equation (8).

3.5. Proximal Support Vector Classification

Following on from the LSVC method, the PSVC approach was developed by (Fung & Mangasarian, 2001b). They also provided a multiclass version of the algorithm (Fung & Mangasarian, 2001a), but using a one-against-all approach instead of a direct all-together extension. The two-class aim is to

$$\text{minimise} \quad \left(\frac{1}{2}\left[\|\mathbf{w}\|_{\mathcal{F}}^2 + b^2\right] + C\sum_{i=1}^{N}\xi_i^2\right) \qquad (19)$$
$$\text{subject to } y_i\left[\langle\Phi(\mathbf{x}_i),\mathbf{w}\rangle_{\mathcal{F}} + b\right] = 1 - \xi_i.$$

which is the same as that for LS-SVC §3.3 except for the b^2 term. This can be reformulated to a multi-category problem resulting in,

$$\text{minimise} \quad \left(\frac{1}{2}\sum_{m=1}^{M-1}\left[\|\mathbf{w}_m\|_{\mathcal{F}}^2 + b_m^2\right] + C\sum_{i=1}^{N}\sum_{\psi\in(\Theta-\vartheta_i)}\xi_{i,\psi}^2\right)$$
$$\text{subject to } \sum_{m=1}^{M-1} v_{\psi,m}(\vartheta_i)\left[\langle\Phi(\mathbf{x}_i),\mathbf{w}_m\rangle_{\mathcal{F}} + b_m\right] = \varepsilon_\psi(\vartheta_i) - \xi_{i,\psi}. \qquad (20)$$

Now, define $\mathbf{v}^{*\prime} = \left[\begin{array}{cccccc}\mathbf{v}_1^{*T}(\vartheta_1) & \ldots & \mathbf{v}_1^{*T}(\vartheta_N) & \mathbf{v}_2^{*T}(\vartheta_1) & \ldots & \mathbf{v}_N^{*T}(\vartheta_N)\end{array}\right]^T$, and with this then, as for LS-SVC the optimisation problem has an exact solution,

$$\boldsymbol{\alpha}' = \left(\mathbf{I} + \mathbf{Z}'^T\mathbf{Z}' + \mathbf{v}^{*\prime}\mathbf{v}^{*\prime T}\right)^{-1}\boldsymbol{\varepsilon}' \qquad (21)$$

where everything is as defined in §3.3 and $\mathbf{b} = \sum_{i=1}^{N}\mathbf{V}(\vartheta_i)\boldsymbol{\alpha}_i$. As before, the final solution takes the form of equation (8).

3.6. Bayes Point Machines

BPMs were introduced by Herbrich et al. (2000) and the ideas can be extended to a multi-category problem. In short they consider what they term *Version Space*, $\mathcal{V}$. In the two-class case this is the region in which a weight vector $\mathbf{w}$ can lie without inducing any classification errors on the training set.

Within version space a uniform distribution is assumed over all possible linear (in feature space) classifiers, h, outside it is assumed zero. The *Bayes point classifier* is then given by

$$h_{bp} = \arg\min_{h\in\mathcal{H}}\mathbb{E}_X\left[\mathbb{E}_{H|\{\mathbf{x}_i,y_i\}_{i=1}^N}\left[\ell\left(h(X),H(X)\right)\right]\right] \qquad (22)$$

where $\ell(\cdot,\cdot)$ is some loss function (typically the zero-one loss function is used) and the inner expectation is over classifiers $H \in \mathcal{H}$. One problem with this definition is that it is not usual that there is any knowledge about P_X and so evaluation of $\mathbb{E}_X$ is impossible. With some assumptions about the form of P_X (see Herbrich et al. (2000) for more) it can, however, be shown that the centre of mass,

$$\mathbf{w}_{cm} = \frac{\mathbb{E}_{\mathbf{w}|\{\mathbf{x}_i,y_i\}_{i=1}^N}[\mathbf{w}]}{\|\mathbb{E}_{\mathbf{w}|\{\mathbf{x}_i,y_i\}_{i=1}^N}[\mathbf{w}]\|} \qquad (23)$$

is a good approximation to $\mathbf{w}_{bp}$. Eventually the problem becomes to identify $\mathcal{V}$, which is some contiguous and convex space, and then to find $\mathbf{w}_{cm}$ given that there is a uniform distribution assumed over the weight vectors in this space.

Note that version space is defined by

$$\mathcal{V} = \{\mathbf{w} : y_i\langle\Phi(\mathbf{x}_i), \mathbf{w}\rangle > 0,\ \|\mathbf{w}\| = 1\}, \tag{24}$$

When considering multiple classes then the condition $y_i\langle\Phi(\mathbf{x}_i), \mathbf{w}\rangle > 0$ becomes $\mathbf{V}^T(\vartheta_i)\mathbf{W}\Phi(\mathbf{x}_i) > \left[\begin{array}{cccc} 0 & 0 & \dots & 0 \end{array}\right]^T$ where the inequality indicates component-wise inequalities, and $\mathbf{W} = \left[\begin{array}{ccc} \mathbf{w}_1 & \dots & \mathbf{w}_{M-1} \end{array}\right]^T$. As a result the version space is given by

$$\begin{aligned}\mathcal{V} = \Big\{(\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{M-1}) : V^T(\vartheta_i)\mathbf{W}\Phi(\mathbf{x}_i) \\ > \left[\begin{array}{cccc} 0 & 0 & \dots & 0 \end{array}\right]^T, \|\mathbf{w}_m\| = 1\ \forall m\Big\},\end{aligned} \tag{25}$$

which is identical form to equation (24). Extensions of the *kernel billiards* algorithm described by Herbrich et al. (2000) can be used to find $\mathbf{W}_{cm}$. Their method for including training errors can also be incorporated.

4. Implementation Issues

The algorithms discussed have all been shown to yield multi-category variants of a very similar form to their binary counterparts. As a result it appears likely that variants of the optimisation procedures used in the two-class cases should be derivable.

The parallels between the two- and multi-category representations means also that those multi-category variants of those algorithms which have been developed directly with optimisation speed considerations in mind (for instance PSVC and LS-SVC) will exhibit similar properties. Similarly properties of other algorithms (*e.g.* the interpretation of ν in $\nu-$SVC) will also carry over.

One continuing issue with many algorithms is however the optimisation speed. It has been found by many authors (Hsu & Lin, 2002a; Fürnkranz, 2002; Rifkin & Klautau, 2004, for example) that pairwise methods can actually be orders of magnitude faster than all-together approaches. To make use of this consider that a series of $\frac{M(M-1)}{2}$ two-class classifiers have been found. It is straightforward to understand these as giving a distance along a vector perpendicular to a class boundary.

For instance the function underlying the classification between classes ϑ and ψ can be understood to be giving a distance along the vector $\mathbf{v}_\psi(\vartheta)$. But note also from equation (8) that $\alpha_{i,\psi}$ is the coefficient of $\mathbf{v}_\psi(\vartheta_i)$ in $\mathbf{f}(\mathbf{x}_i)$. Thus it turns out that having performed pairwise classification then setting the value $\alpha_{i,\psi}$ equal to α_i, as found when considering classes ψ and ϑ_i is an efficient *ad hoc* method of mapping the two-class outputs to the all-together geometric structure.

An all-together algorithm can now be used to *fine-tune* the result. We present a SVC example of this to give an idea of the general approach. The dataset used was obtained from the University of California repository (Blake & Merz, 1998). It is a DNA dataset consisting of three classes 'Intron-Extron'; '*IE*', 'Extron-Intron'; '*EI*', and 'Neither'; '*N*'. Training and test results are presented in Figure 3. Here it is clear to see how

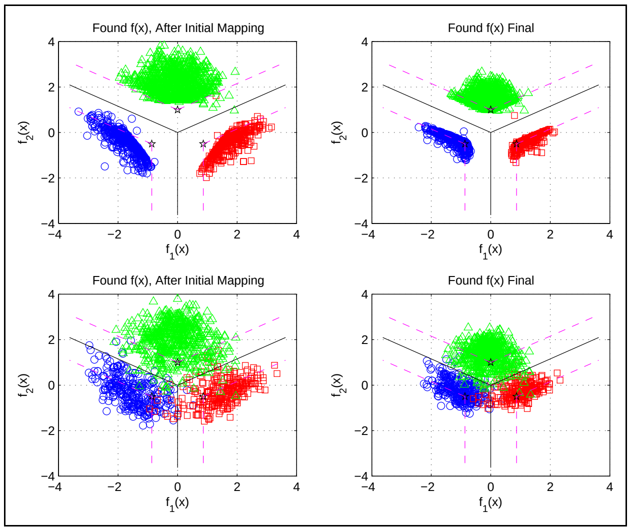

Figure 3. DNA data outputs for training and test data cases. *The mapped pairwise and optimised 'all-together' results are shown. Margins analogous to those in the two-class case are shown by dashed lines. Training data forms the top row, test data the bottom. The 'N' case is given by green triangles, 'EI' by blue circles, and 'IE' by red squares.*

the pairwise result (the left two Subfigures) has been mapped into the classification plane of Figure 1, and what changes are made in performing the 'all-together' additional optimisation (the right two Subfigures). In short the '*N*' class appears to have intermingled a little more with the '*EI*' class and less with the '*IE*' class. As well the 'all-together' outputs fill the corners of the margin intersections more completely, while the pairwise outputs tend to cut them off.

5. Experiments

Extensive investigations into comparative performance of multi-category SVM methods have been detailed by Hsu and Lin (2002a), again on datasets from Blake and Merz (1998), and they present current benchmark training times. As discussed, their work has found that pairwise coupling approaches are far more computationally efficient than others. In performing comparative work a standard binary, and a multi-category version of the Sequential Minimal Optimisation (SMO) algorithm as detailed by Hill and Doucet (2005) was used, and these were coded in a straightforward way. No dynamic caching or low-level code refinements were used in this initial proof-of-concept investigation as it was felt that such detailed optimisations are best done together in a consistent way, as in the dedicated comparative work of Hsu and Lin (2002a).

The implementation time was found to be heavily dependent on the tolerance to within which convergence is desired (referred to as τ by Keerthi et al. (2001)). The effect of this has been investigated for two values of τ, and the results are tabulated in Table 1. In these experiments Gaussian kernels were used and values of σ and C were chosen by trial and error such that output accuracies (accuracy is the percentage classification error rate) of the 'all-together' implementation were comparable to those of Hsu and Lin (2002a).

The actual accuracies recorded are given in the Table, however recall that, as mentioned in Section 3, the optimisation problem being solved is the generic SVC 'all-together' one and, as such, judicious choices of σ and C should mean that *the same accuracy rates are achievable by all such algorithms.* Clearly as the implicit model behind the pairwise approach is slightly different it may indeed be able to achieve slightly different accuracy results. With this in mind the aim here has not been to incessantly tweak hyperparameters to achieve marginally superior results, but simply to look at the big picture of performance.

In continuing with this mindset, no class weightings were introduced, and target vectors were set to be equidistant. It may well be the case that these could actually be perturbed, and class weights used to improve performance, with no additional computational effort, however in this initial work this was not done.

The experiments were all run on a 2.8GHz P4 with 1GB RAM[3]. From Table 1 it becomes apparent that the optimisation times for two approaches presented are of magnitudes similar to those of Hsu and Lin, when the difference in computing power is taken into account. Although it has not been the aim of this work to produce highly refined optimal code, and while such comparisons are always going to be problematic in terms of implementation specifics, this result is, in itself, positive. Generally, when using only the 'all-together' methodology with $\tau = 0.001C$ convergence times are similar to those of Hsu and Lin for their 'all-together' implementation. Briefly, their optimisation times were; for DNA, 13.5s, for vehicle 88.6s, for satimage 48.2s, for segment 66.4s, for vowel 14.1s, and for letter 8786.2s. As such we consider the advantage obtained here through extra computational power as roughly equivalent to the effect of their extra coding.

[3]Hsu and Lin (2002a) had a 500MHz P3 with 384MB RAM.

While clearly very approximate this does however help to demonstrate the relative effect of combining the pairwise and 'all-together' algorithms in context. In short it typically halves them, although the variation on this is quite large. This result is consistent for both values of τ, as is the result that error rate results do not strongly favour the pairwise or 'all-together' methods; this is always going to be a case-by-case issue.

6. Conclusion

The geometric construction detailed in Section 2 can be considered an effective tool for understanding and implementing multi-category classification. It has been shown in Section 3 to extend many existing two-class methodologies in a straightforward way while preserving attractive properties of individual algorithms. It also provides an effective mechanism for incorporating the results of pairwise classification as a means of reducing training times when aiming to achieve an all-together result.

References

Allwein, E. L., Schapire, R. E., & Singer, Y. (2001). Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, *1*.

Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

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

Dietterich, T., & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, *2*, 263–286.

Fung, G., & Mangasarian, O. L. (2001a). *Multicategory proximal support vector machine classifiers* (Technical Report 01-06). Data Mining Institute.

Fung, G., & Mangasarian, O. L. (2001b). Proximal support vector machine classifiers. *Proceedings KDD-2001* (pp. 77–86). San Francisco.

			$\tau = 0.03C$					$\tau = 0.001C$				
Problem	M	N	Pair	All	Alone	ER(P)	ER(A)	Pair	All	Alone	ER(P)	ER(A)
DNA	3	2000	0.8	1.1	1.5	4.4	4.6	1.1	3.7	11.7	4.6	4.5
Vehicle	4	766	0.4	2.7	5.3	15.0	18.8	0.5	3.5	3.9	17.5	20.0
Satimage	6	4435	3.0	10.8	41.8	10.6	10.8	3.6	9.0	27.6	9.7	9.2
Segment	7	2079	2.4	13.2	47.9	3.0	2.6	3.2	16.2	42.0	3.0	3.0
Vowel	11	891	0.7	3.5	13.3	3.0	3.0	1.0	18.5	22.8	3.0	3.0
Letter	26	15000	129.0	129.9	2119.2	8.8	8.8	142.3	1373.7	5573.4	8.9	8.8

Table 1. Optimisation times (seconds) and error rates (percentage) for various example problems. *Columns present experimentally obtained results using the pairwise, 'all-together' multi-category SMO algorithm discussed. In all cases 'Pair' refers to pairwise optimisation time results, 'All' to additional refinement time, and 'Alone' to the 'all-together' algorithm without pairwise initialisation optimisation time results. Meanwhile 'ER(P)' refers to the test error rate of the pairwise method and 'ER(A)' to that of the 'all-together' algorithm.*

Fürnkranz, J. (2002). Round robin classification. *Journal of Machine Learning*, *2*, 721–747.

Herbrich, R., Graepel, T., & Campbell, C. (2000). Bayes point machines. *Journal of Machine Learning Research.*

Hill, S. I., & Doucet, A. (2005). *A framework for kernel-based multi-category classification* (Technical Report CUED/F-INFENG/TR.508). University of Engineering, Cambridge University.

Hsu, C.-W., & Lin, C.-J. (2002a). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, *13*, 415–425.

Hsu, C.-W., & Lin, C.-J. (2002b). A simple decomposition method for support vector machines. *Journal of Machine Learning*, *46*, 291–314.

Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., & Murthy, K. R. K. (2001). Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation*, *13*, 637–649.

Kreßel, U. H.-G. (1999). Pairwise classification and support vector machines. In B. Schölkopf, C. J. C. Burges and A. J. Smola (Eds.), *Advances in kernel methods: Support vector learning.* MIT Press.

Lee, Y., Lin, Y., & Wahba, G. (2004). Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, *99*, 659–672.

Mangasarian, O. L., & Musicant, D. R. (2001). Lagrangian support vector machines. *Journal of Machine Learning Research*, *1*, 161–177.

Rifkin, R., & Klautau, A. (2004). In defense of one-vs-all classification. *Journal of Machine Learning Research*, *5*, 101–141.

Schölkopf, B., & Smola, A. J. (2002). *Learning with kernels.* MIT Press.

Van Gestel, T., Suykens, J. A. K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., De Moor, B., & Vandewalle, J. (2001). Benchmarking least squares support vector machine classifiers. *Machine Learning*, *54*, 5–32.

Van Gestel, T., Suykens, J. A. K., Lanckriet, G., Lambrechts, A., De Moor, B., & Vandewalle, J. (2002). Multiclass LS-SVMs: Moderated outputs and coding-decoding schemes. *Neural Processing Letters*, *15*, 45–58.

Vapnik, V. N. (1998). *Statistical learning theory.* Wiley.

Weston, J. A. E., & Watkins, C. (1999). Support vector machines for multi-class pattern recognition. *Proceedings of the 7th European Symposium On Artificial Neural Networks.*