A Novel Framework for Multi-class Classification via Ternary Smooth Support Vector Machine

Chih-Cheng Chang, Li-Jen Chien [1], Yuh-Jye Lee

Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology
Taipei, 106 Taiwan

Abstract

This paper extends the previous work in smooth support vector machine (SSVM) from binary to k-class classification based on a single machine approach and call it multi-class smooth SVM (MSSVM). This study implements MSSVM for a ternary classification problem and labels it as TSSVM. For the case $k > 3$, this study proposes a one-vs.-one-vs.-rest (OOR) scheme that decomposes the problem into $k(k-1)/2$ ternary classification subproblems based on the assumption of ternary voting games. Thus, the k-class classification problem can be solved via a series of TSSVMs. The numerical experiments in this study compare the classification accuracy for TSSVM/OOR, one-vs.-one, one-vs.-rest schemes on nine UCI datasets. Results show that TSSVM/OOR outperforms the one-vs.-one and one-vs.-rest for all datasets. This study includes further error analyses to emphasize that the prediction confidence of OOR is significantly higher than the one-vs.-one scheme. Due to the nature of OOR design, it can detect the hidden (unknown) class directly. This study includes a "leave-one-class-out" experiment on the `pendigits` dataset to demonstrate the detection ability of the proposed OOR method for hidden classes. Results show that OOR performs significantly better than one-vs.-one and one-vs.-rest in the hidden class detection rate.

Key words: confidence, hidden classes, multi-class classification, smooth method, support vector machine, ternary voting games

[1]Corresponding author.

1. Introduction

Support Vector Machines (SVMs) [5, 8, 11] have recently become promising learning algorithms for binary classification. How to effectively extend binary SVMs to solve multi-class classification problems remains a topic of research. There are two most popular approaches to deal with k-class classification problems. One is the single machine approach, which generates a sophisticated classifier by solving a huge and complex optimization problem [4, 10, 23, 33, 34]. The other is the multiple-machine approach, which separates a multi-class classification problem into a series of smaller binary classification tasks [1, 3, 10, 12, 15, 16, 19, 20, 21, 22, 30].

Previous studies discuss and compare these two variant multi-class SVM [17, 31]. Comparison results suggest that two simple and well-known multiple-machine approaches, the one-vs.-rest scheme [3] and the one-vs.-one scheme [15, 21, 22], should be used because of their clarification, good accuracy reports, and decreased computation cost.

This paper investigates the one-vs.-rest scheme and the one-vs.-one scheme and addresses some concerns on them. The one-vs.-rest scheme builds up k binary subclassifiers, training the ith subclassifier f_i from data in class i and the remaining classes for $j = \{1, 2, \ldots, k\} \setminus \{i\}$. This scheme assigns the class label for a new instance x based on the largest value of $f_i(x)$. However, using the one-vs.-rest approach for the largest value strategy is debatable. That is, dependently comparing k output values among $f_1, f_2, \ldots, f_k$ that are trained independently might not be reasonable. Since each classifier is trained by an unbalanced binary classification problem, it tends to produce a negative output for all classifiers. Conversely, the one-vs.-one scheme constructs $k(k-1)/2$ binary subclassifiers. This approach trains each subclassifier f_{ij} from data in class i and class j, where $i, j \in \{1, 2, \ldots, k\}$ and $i < j$. If f_{ij} says x is in class i, it increases the vote for class i by one; if not, it increases the vote for class j by one. After x goes through all $k(k-1)/2$ subclassifiers, the one-vs.-one scheme applies the majority vote strategy [15] for labeling x to the class with the most votes. If a given x with a known label is irrelevant to two particular classes under a present f_{ij}, the f_{ij} still must make a binary decision, voting on an unrelated class due to the mechanism of the one-vs.-one scheme. This type of vote is a nuisance vote. The one-vs.-one scheme generates many of nuisance votes in binary voting games. Numerous nuisance votes pollute the clarity of the one-vs.-one scheme in classifying x into a correct class, decreasing prediction accuracy and confidence.

To address the above concerns of one-vs.-rest and one-vs.-one, this study introduces a multi-class smooth support vector machine (MSSVM) formulation. The proposed approach extends previous work in smooth support vector machine (SSVM) [25] from binary to k-class classification using the single machine approach [9, 10]. Training an MSSVM involves solving a huge and complicated unconstrained minimization problem, especially when k is large. In many cases, this prohibits the practical application of MSSVM. This study implements MSSVM for a ternary classification problem , calling it TSSVM. For the case $k > 3$, we propose a novel scheme, one-vs.-one-vs.-rest (OOR). This scheme decomposes the problem into $k(k-1)/2$ ternary classification subproblems based on the assumption of ternary voting games [14]. This makes it possible to solve the k-class classification problem via a series of TSSVMs. The OOR scheme is able to filter out lots of nuisance votes, increasing both prediction accuracy and confidence. The experiments in this study, we compare accuracy rates for TSSVM/OOR, one-vs.-one, one-vs.-rest, and the popular LIBSVM (one-vs.-one) [6] on nine UCI datasets. This paper uses 2-norm soft margins to measure the training errors in other schemes. For completeness and comparison purposes, this study lists the numerical results of LIBSVM which implements the 1-norm soft margin SVM and is different from our 2-norm soft margin SSVM, as the base classifier. The numerical results show that TSSVM/OOR outperforms the one-vs.-one and one-vs.-rest for all the datasets, and it is comparable to LIBSVM as well. Detailed error analysis shows that the prediction confidence of OOR is higher than the one-vs.-one scheme.

In many pattern recognition applications, it is necessary to provide the detection ability to identify hidden classes that do not appear in the collection of training data [35]. Due to the nature of OOR design, it can detect the hidden (unknown) class directly. To demonstrate the detection ability of the proposed OOR method for hidden classes, this study includes a “leave-one-class-out” experiment on the `pendigits` dataset. The results show that OOR significantly outperforms the one-vs.-one and one-vs.-rest schemes in the hidden class detection rate, and achieves slightly better testing accuracy.

The following briefly describes some notations used in this paper. A training dataset is $\{(x^i, y_i)\}_{i=1}^m$, where $x^i \in R^{n\times 1}$ is the ith training instance and $y_i \in \{1, 2, \ldots, k\}$ is the class label of x^i. For notational convenience, the training dataset is rearranged as an $m \times n$ matrix A, and $A_i = (x^i)'$ corresponds to the ith row of A. The notation of $\langle x, w\rangle$ denotes the inner product of x and w, where both x and w belong to $R^{n\times 1}$. Column vectors of ones and zeros are denoted by bold **1** and **0** respectively.

The rest of this paper is organized as follows. Section 2 provides a brief introduction of SSVM and reduced kernel trick in SVM. Section 3 describes the formulation of MSSVM and the implementation of TSSVM and OOR. Section 4 presents the comparison results of classification accuracy, confidence, purification ability and detection ability for hidden classes. Section 5 concludes the paper.

2. Smooth Support Vector Machine for Binary Classification

In the last decade, researchers have proposed many different formulations of SVMs. However, all of these variants solve either a constrained minimization problem or an unconstrained minimization problem. The objective function in such optimization problems consists of two main parts: a miss-classification penalty part that stands for *model bias*, and a regularization part that controls the *model variance*. As the bases of MSSVM, we briefly introduce the smooth support vector machine [25], the kernel trick to extend the linear SVM to the nonlinear one, and reduced kernel technique. The reduced kernel technique overcomes the computational difficulties that occur in dealing with a fully dense square kernel matrix in the conventional nonlinear SVM.

2.1. Smooth Support Vector Machine

The aim of a SVM is to identify the separating hyperplane with the largest margin in the training data. This allows the SVM classifiers to have a better generalization ability in predicting the label for a new unseen instance. This hyperplane is "optimal" in the sense of statistical learning [33].

Begin with linear SVM for the binary classification. Given a training dataset $S = \{(x^1, y_1), \ldots, (x^m, y_m)\} \subseteq R^n \times R$, where $x^i \in R^n$ is the input data and $y_i \in \{-1, 1\}$ is the corresponding class label. Solving a convex optimization problem generates a conventional SVM separating hyperplane as follows:

$$\begin{array}{ll} \min\limits_{(w,b,\xi)\in R^{n+1+m}} & C\sum_{i=1}^m \xi_i + \frac{1}{2}\|w\|_2^2 \\ \text{subject to} & y_i(\langle x^i, w\rangle + b) + \xi_i \geq 1 \\ & \xi_i \geq 0, \quad \text{for } i = 1, 2, \ldots, m, \end{array} \tag{1}$$

where C is a positive parameter controlling the trade-off between the training error (model bias) and the part of maximizing the margin (model variance) that is achieved by minimizing $\|w\|_2^2$. Unlike the conventional SVM in (1), the smooth support vector machine minimizes the square of the slack vector ξ with weight $\frac{C}{2}$. The SSVM-methodology appends $\frac{b^2}{2}$ to the term being minimized. This expansion leads to the following minimization problem:

$$\begin{array}{rl} \min\limits_{(w,b,\xi)\in R^{n+1+m}} & \frac{C}{2}\sum_{i=1}^{m}\xi_i^2 + \frac{1}{2}(\|w\|_2^2 + b^2) \\ \text{subject to} & y_i(\langle x^i, w\rangle + b) + \xi_i \geq 1 \\ & \xi_i \geq 0, \quad \text{for } i = 1, 2, \ldots, m. \end{array} \tag{2}$$

In the solution of (2), ξ is given by $\xi_i = \{1 - y_i(\langle x^i, w\rangle + b)\}_+$ for all i where the *plus* function x_+ is defined as $x_+ = \max\{0, x\}$. Thus, we can replace ξ_i in (2) by $\{1 - y_i(\langle x^i, w\rangle + b)\}_+$. This converts the problem (2) into the following unconstrained minimization problem:

$$\min_{(w,b)\in R^{n+1}} \frac{C}{2}\sum_{i=1}^{m}\{1 - y_i(\langle x^i, w\rangle + b)\}_+^2 + \frac{1}{2}(\|w\|_2^2 + b^2). \tag{3}$$

This formulation reduces the number of variables from $n + 1 + m$ to $n + 1$. However, because the objective function to be minimized is not twice differentiable, it precludes the use of a fast Newton method. In the SSVM, the plus function x_+ is approximated by a smooth *p-function*, $p(x, \alpha) = x + \frac{1}{\alpha}\log(1 + e^{-\alpha x}), \alpha > 0$. Replacing the plus function with an accurate smooth approximation allows the p-function to give the following smooth support vector machine formulation:

$$\min_{(w,b)\in R^{n+1}} \frac{C}{2}\sum_{i=1}^{m} p(\{1 - y_i(\langle x^i, w\rangle + b)\}, \alpha)^2 + \frac{1}{2}(\|w\|_2^2 + b^2), \tag{4}$$

where $\alpha > 0$ is the smooth parameter. The objective function in problem (4) is strongly convex and infinitely differentiable. Hence, it has a unique solution and can be solved using a fast Newton-Armijo algorithm. For the nonlinear case, Mercer kernel mapping can construct a linear classifier in the feature space. This formulation can be extended to the nonlinear SVM as using a generalized support vector machine (GSVM) [27]:

$$\min_{(v,b)\in R^{m+1}} \frac{C}{2}\sum_{i=1}^{m} p([1 - y_i\{\sum_{j=1}^{m} v_j K(x^i, x^j) + b\}], \alpha)^2 + \frac{1}{2}(\|v\|_2^2 + b^2), \tag{5}$$

where $K(x^i, x^j)$ is a kernel function. This kernel function represents the inner product of $\phi(x^i)$ and $\phi(x^j)$, where ϕ is a certain nonlinear mapping from input space R^n to a feature space $\mathcal{F}$. We do not need to know the mapping of ϕ explicitly. This is the so-called kernel trick. The most popular kernel is called Gaussian Kernel (Radial Basis Function), defined as

$$K(x^i, x^j) = e^{-\gamma\|x^i - x^j\|_2^2}. \tag{6}$$

The nonlinear SSVM classifier $f(x)$ can be expressed as follows:

$$f(x) = \sum_{v_j \neq 0} v_j K(x^j, x) + b. \tag{7}$$

This classifier is a linear combination of basis functions $\{1\} \bigcup \{K(x^j, \cdot)\}_{j=1}^{m}$. The coefficient v_j is determined by solving an optimization problem (5) and the data points with corresponding nonzero coefficients are called support vectors. It is often desirable to have fewer support vectors.

2.2. Reduced Support Vector Machine

In large scale problems, the fully dense kernel matrix creates space and time complexity problems when dealing with (5). To avoid these difficulties and reduce model complexity, this study introduces the reduced kernel technique [24]. This technique reduces the big full kernel matrix without sacrificing its generalization ability. The key idea of the reduced kernel technique is to randomly select a small portion of data as the kernel bases and then generate a thin rectangular kernel matrix to replace the original square full kernel matrix. Based on the low-rank Nyström approximation [32, 36], the full kernel matrix can be approximated as:

$$K(A, A') \approx K(A, \bar{A}')K(\bar{A}, \bar{A}')^{-1}K(\bar{A}, A'), \tag{8}$$

where $K(A, A') = K_{m\times m}$, $\bar{A}_{\bar{m}\times n}$ is a subset of A and $K(A, \bar{A}') = \bar{K}_{m\times\bar{m}}$ is a reduced kernel. Thus, we have

$$K(A, A')v \approx K(A, \bar{A}')K(\bar{A}, \bar{A}')^{-1}K(\bar{A}', A)v = K(A, \bar{A}')\bar{v}, \tag{9}$$

where $\bar{v} \in R^{\bar{m}}$ is an approximated solution of v via the reduced kernel technique. The reduced kernel method constructs a compressed model and reduces the computational cost from $\mathcal{O}(m^3)$ to $\mathcal{O}(\bar{m}^3)$. The solution with reduced kernel matrix approximates the solution with the full kernel matrix and the performance with experiments in [24, 26].

3. The Framework for Multi-class Classification via TSSVM

Following the derivation of SSVM and RSVM, this study expands the SSVM methodology from binary to k-class classification and give the formulation of MSSVM. This section describes the implementation of TSSVM and the framework of the OOR scheme in detail.

3.1. Multi-class Smooth Support Vector Machine

For the multi-class classification problems, this section expands the idea of comparing k margin requirements under a single optimization problem [34] and formulate the problem as follows:

$$\begin{array}{rl} \min\limits_{(w,b,\xi)\in R^{k(n+1+m)-m}} & \frac{1}{2}\sum\limits_{j=1}^{k}(\|w^j\|_2^2+b_j^2)+\frac{C}{2}\sum\limits_{i=1}^{m}\sum\limits_{j\neq y_i}\xi_{ij}^2 \\ \textit{subject to}: & \langle x^i, w^{y_i}\rangle + b_{y_i} \geq \langle x^i, w^j\rangle + b_j + 1 - \xi_{ij} \\ & i=1,\ldots,m,\ j\in\mathscr{C}\backslash y_i,\ \mathscr{C}=\{1,\ldots,k\}. \end{array} \tag{10}$$

Similar to the derivation of SSVM in Section 2.1, we can obtain an unconstrained minimization formulation for the linear MSSVM:

$$\begin{array}{rl} \min\limits_{(w,b)\in R^{k(n+1)}} & \frac{1}{2}\sum\limits_{j=1}^{k}(\|w^j\|_2^2+b_j^2)+ \\ & \frac{C}{2}\sum\limits_{i=1}^{m}\sum\limits_{j\neq y_i} p([1-\langle x^i, w^{y_i}-w^j\rangle-(b_{y_i}-b_j)],\alpha)^2. \end{array} \tag{11}$$

Because the objective function in problem (11) is twice differentiable, the Newton-Armijo algorithm can be applied here again, and the unique solution of (11) approaches the unique solution of the equivalent optimization problem (10) when the smoothing parameter α approaches infinity. The proof can also be adapted from the results in binary SSVM [25]. Moreover, because linear MSSVM only needs to find $k(n+1)$ variables in problem (11), and the number of variables is much smaller than other single machine approaches.

Applying the kernel trick as in Section 2.1, it is easy to modify the linear MSSVM (11) to obtain a nonlinear multi-class smooth SVM:

$$\begin{array}{rl} \min\limits_{(v,b)\in R^{k(m+1)}} & \frac{1}{2}\sum\limits_{j=1}^{k}(\|v^j\|_2^2+b_j^2)+ \\ & \frac{C}{2}\sum\limits_{i=1}^{m}\sum\limits_{j\neq y_i} p([1-\langle K(A,x^i), v^{y_i}-v^j\rangle-(b_{y_i}-b_j)],\alpha)^2. \end{array} \tag{12}$$

This optimization problem (12) can generate a highly nonlinear separating surface while retaining the strong convexity and differentiability properties for any arbitrary kernel. The Newton-Armijo algorithm is also applicable to this problem.

Equation (12) must find $k(m+1)$ variables, and its computational complexity is $\mathcal{O}((k(m+1))^3)$. Like other single machine approaches, nonlinear MSSVM becomes impractical when confronting large-scale data problems with either a large m or k. To overcome these computational difficulties, the proposed approach alleviates the difficulty of a large m. This study uses the same idea of RSVM in Section 2.2 to replace the full kernel $K(A,x)$ in the problem (12) with a reduced kernel $K(\bar{A},x)$, where $\bar{A} \in R^{\bar{m}\times n}$ is a random subset of A and $\bar{m} \ll m$. This produces a nonlinear MSSVM with reduced kernel as follows:

$$\begin{aligned}\min_{(\bar{v},b)\in R^{k(\bar{m}+1)}} \quad & \tfrac{1}{2}\sum_{j=1}^{k}(\|\bar{v}^j\|_2^2 + b_j^2)+ \\ & \tfrac{C}{2}\sum_{i=1}^{m}\sum_{j\neq y_i} p([1-\langle K(\bar{A},x^i),\bar{v}^{y_i}-\bar{v}^j\rangle-(b_{y_i}-b_j)],\alpha)^2.\end{aligned} \tag{13}$$

For the optimization problem (13), we only need to find $k(\bar{m}+1)$ variables. Thus, this approach efficiently mitigates computational cost and memory usage without sacrificing prediction accuracy.

3.2. Ternary Smooth Support Vector Machine (TSSVM)

Although MSSVM has been developed for any ($k \geq 3$)-class classification problems, it is still too complicated to implement when k is large. We only implement the ternary smooth support vector machine (TSSVM) and use it as the base classifier of our framework for multi-class classification. For $k > 3$, this study proposes the OOR scheme introduced in Section 3.3. This section shows the implementation of TSSVM. Based on the optimization problem (11), the TSSVM formulation is as follows:

$$\min_{\omega\in R^{3(n+1)}} \quad \Phi_\alpha(\omega) = \tfrac{1}{2}\|\omega\|_2^2 + \tfrac{C}{2}\|p(\mathbf{1}-B\omega,\alpha)\|_2^2, \tag{14}$$

where ω and B are

$$\omega = \begin{bmatrix} w^1 \\ b_1 \\ w^2 \\ b_2 \\ w^3 \\ b_3 \end{bmatrix}_{3(n+1)\times 1}, \quad B = \begin{bmatrix} A^1 & \mathbf{1}^1 & -A^1 & -\mathbf{1}^1 & O^1 & \mathbf{0}^1 \\ A^1 & \mathbf{1}^1 & O^1 & \mathbf{0}^1 & -A^1 & -\mathbf{1}^1 \\ -A^2 & -\mathbf{1}^2 & A^2 & \mathbf{1}^2 & O^2 & \mathbf{0}^2 \\ O^2 & \mathbf{0}^2 & A^2 & \mathbf{1}^2 & -A^2 & -\mathbf{1}^2 \\ -A^3 & -\mathbf{1}^3 & O^3 & \mathbf{0}^3 & A^3 & \mathbf{1}^3 \\ O^3 & \mathbf{0}^3 & -A^3 & -\mathbf{1}^3 & A^3 & \mathbf{1}^3 \end{bmatrix}_{2m\times 3(n+1)},$$

where O^j is a $m_j \times n$ zero matrix, and both $\mathbf{1}^j$ and $\mathbf{0}^j$ are $m_j \times 1$ vectors for $j = 1, 2, 3$.

Nonlinear cases simply require changing the training inputs from A to $K(A, A')$ before constructing the matrix B.

When using the Newton-Armijo algorithm to solve the optimization problem (14), the most important ingredients are the gradient and the Hessian matrix of $\Phi_\alpha(\omega)$. Apply the chain rule from calculus and some elementary matrix algebra to get the following formulae:

$$\lim_{\alpha\to\infty} \nabla\Phi_\alpha(\omega) = [\omega - CB'(\mathbf{1} - B\omega)_+] \tag{15}$$

and

$$\lim_{\alpha\to\infty} \nabla^2\Phi_\alpha(\omega) = [I + CB'diag(S_\infty(\mathbf{1} - B\omega))B]\,, \tag{16}$$

where

$$S_\infty(x) = \lim_{\alpha\to\infty} \frac{1}{1+e^{-\alpha x}} = \frac{1+sign(x)}{2}. \tag{17}$$

In fact, the limit of the gradient of $\Phi_\alpha(\omega)$ as α goes to infinity (15) is the gradient of (11), and the limit of the Hessian matrix of $\Phi_\alpha(\omega)$ as α approaches infinity (16) is an element of the generalized Jacobian of (11) [7]. Although the Newton-Armijo algorithm is globally convergent with any initial solution, a wise choice of the initial solution can significantly reduce the computation time. To speed up the convergence, use the solution of one-vs.-rest as the initial setting of ω^0 in the Newton-Armijo algorithm.

3.3. One-vs.-One-vs.-Rest Strategy

To deal with the case for $k > 3$ in k-class classification problem and filter out the nuisance votes occurring in the one-vs.-one scheme, this section proposes a one-vs.-one-vs.-rest (OOR) scheme that decomposes a k-class classification problem into $k(k-1)/2$ subproblems. Each subproblem is a ternary classification problem that consists of two particular classes and the rest of the classes fused as the third class from the training set. The TSSVM classifier developed in the previous section can serve as a tool to generate the $k(k-1)/2$ ternary classifiers. For a new instance x, each subclassifier has three options to vote for: each of the two particular classes and the rest-class, respectively. After x goes through all subclassifiers of OOR, like one-vs.-one, OOR determines the class label of x using the majority voting strategy except for the rest-class.

The OOR scheme can reasonably assign the class label based on the assumption of ternary voting games [14]. If a given x has no relation with two particular classes under a present subclassifier, the subclassifier can abstain in this round of ternary voting games by voting for the rest-class. The one-vs.-one scheme does not have the merit of this option. Thus, the one-vs.-one scheme suffers from the restriction of binary subclassifiers, and each subclassifier can only make the decision about one pair of binary choices at a time. This approach inevitably generates many nuisance votes. A plethora of nuisance votes pollutes the final decision of the majority strategy, decreasing prediction accuracy and confidence. An interesting phenomenon that happens to the prediction results of one-vs.-one is that the number of votes in the champion class is always close to the runner-up, whereas OOR is able to filter out the majority of nuisance votes, clear up the tense and unstable circumstance, and simultaneously increase prediction accuracy and confidence. Sections 4.1 to 4.3 present these numerical results. Furthermore, the OOR design has an excellent ability to detect the hidden classes that do not appear in the existing classes of the training data. The demonstration of the detection ability of OOR for hidden classes will be shown in Section 4.4 as well.

To visualize the effectiveness of the OOR scheme in generating a highly nonlinear separating surface, this study tests two synthetic nine-block datasets, which are quaternary classification problems. Each block includes 150 training points without and with noises on the boundaries, and the distinct color means the different category. To compare the performance of OOR, one-vs.-one, and one-vs.-rest, we uniformly generate 40000 testing points in the nine-block box and apply the Gaussian kernel (6) to all of the classification methods. Figure 1 and Figure 2 show the training sets and testing results. These results clearly show that OOR generates a highly nonlinear separating surface, and its accuracy rates are higher than the other methods. The following section tests TSSVM/OOR on some real datasets from UCI.

4. Experimental Results

This study presents experimental results in four parts. Section 4.1 tests TSSVM/OOR, one-vs.-one, one-vs.-rest, and LIBSVM [6] on nine UCI datasets to determine their accuracy. LIBSVM is a popular SVM solver, implemented as 1-norm soft margin SVM. This paper uses a 2-norm soft margin SSVM as the underlying classifier. To compare the performance between the two

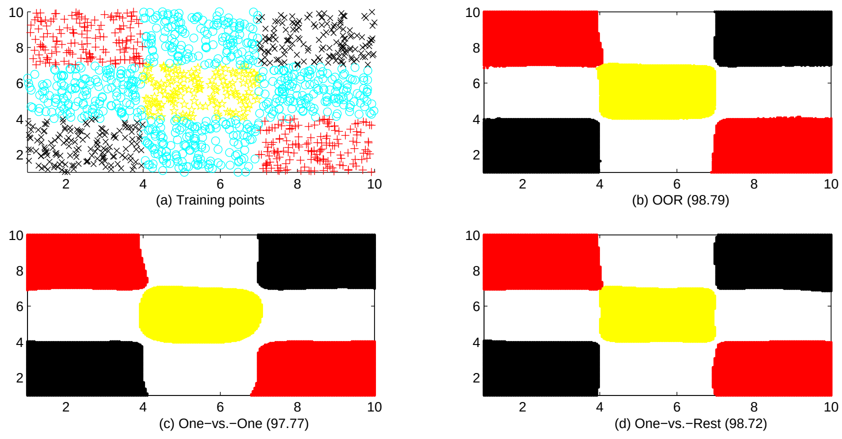

Figure 1: Nine-block dataset without boundary noises. The accuracy rates are also listed in the parentheses.

different forms of SVM classifier, we present all numerical results for completeness and comparison purpose. Through detailed voting analyses, Section 4.2 shows that OOR is a powerful scheme that can precisely filter out most nuisance votes. Section 4.3 presents further detailed error analysis, showing that OOR achieves higher prediction confidence than one-vs.-one. Section 4.4 demonstrates the detection ability of OOR for hidden classes. All the codes in this study were written in Matlab [28] and all the experiments were executed in the same environment. The computational configuration was a P4 2.0GHz computer with 1GB of memory and Windows XP operating system.

4.1. Numerical Results on UCI Datasets

TSSVM was first tested on four ternary classification problems, including `iris`, `tae`, `wine`, and `dna`, to prove its usability. OOR was tested on `glass`, `segment`, `image`, `satimage`, and `pendigits`. These nine datasets are public and available either on UCI repository machine learning database [2] or UCI statlog collection [29]. Table 1 lists the characterization of these datasets.

In implementation, all the training datasets were normalized to have mean 0 and variance 1. The same scaling was applied to the testing datasets

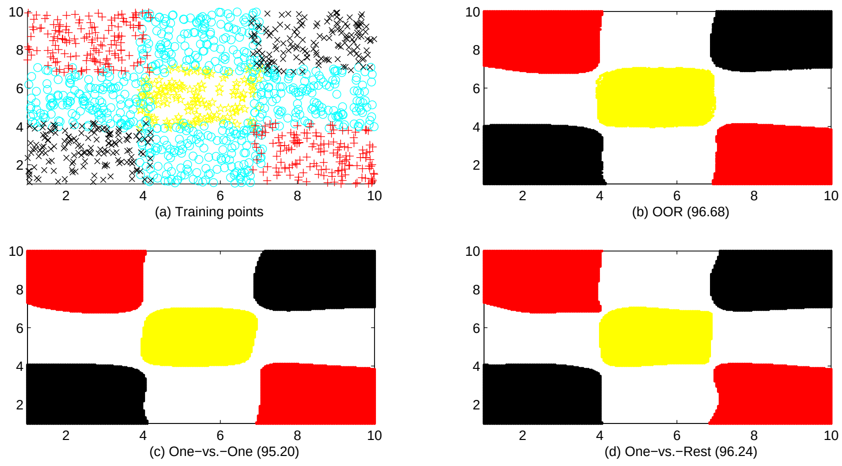

Figure 2: Nine-block dataset with boundary noises. The accuracy rates are also listed in the parentheses.

Table 1: Data characterization

datasets	#training data	#testing data	#classes	#attributes
iris	150	0	3	4
tae	151	0	3	5
wine	178	0	3	13
dna	2000	1186	3	180
glass	214	0	6	9
segment	2310	0	7	18
image	210	2100	7	19
satimage	4435	2000	6	36
pendigits	7494	3498	10	16

according to the scale of the training datasets. For datasets without a test set, 10-fold cross validation (CV) was used to evaluate their performance. For detailed description, we used two nested CV loops; the outer loop estimated the generalization error while the inner CV tuned the parameters. This approach ensures that the validation set in outer loop is independent of model selection. The Gaussian kernel (6) was applied to all multi-class SVMs during both training and testing. The subclassifiers of a multiple-machine approach share a common pair of learning parameters of (C, γ). Note that SSVM is applied to construct one-vs.-one and one-vs.-rest schemes, and TSSVM is employed for the OOR scheme. To reduce computational cost, we apply the reduced kernel technique to TSSVM/OOR and one-vs.-rest when training from `dna`, `segment`, `satimage`, and `pendigits`. To find a desirable pair of (C, γ), a model selection method called nested uniform designs (nested-UDs) [18] is utilized. It only tries on 21 different pairs of (C, γ), which is wisely chosen by number-theoretic methods [13]. The results by using nested-UDs are usually good enough with much less computational cost as compared to the grid-search model selection method. Table 2 reports the best learning parameters for each multi-class SVM on each data problem. Based on the accuracy results listed in Table 3, the TSSVM/OOR approach performs better than one-vs.-one and one-vs.-rest for all datasets, and is mostly superior to LIBSVM as well.

Table 2: Learning parameters of (C, γ)

datasets	TSSVM/OOR	one-vs.-one	one-vs.-rest	LIBSVM
`iris`	(3.16E+3, 1.07E-2)	(3.16E+3, 1.07E-2)	(4.22E+4, 4.17E-3)	(7.50E-1, 3.41E-1)
`tae`	(1.33E+4, 5.25E-1)	(1.33E+4, 5.25E-1)	(1.33E+4, 5.25E-1)	(2.37E+3, 2.04E-1)
`wine`	(1.33E+1, 1.35E-3)	(4.22E-1, 2.29E-2)	(3.16E+3, 1.85E-3)	(1.00E+0, 3.50E-2)
`dna`	(2.37E+2, 2.53E-4)	(3.16E+3, 9.85E-5)	(3.16E+5, 3.47E-4)	(1.00E+4, 8.00E-2)
`glass`	(4.64E+1, 1.25E-1)	(1.00E+1, 5.53E-2)	(1.00E+4, 2.52E-2)	(3.16E+3, 4.50E-3)
`segment`	(1.00E+3, 3.82E-2)	(2.37E+2, 4.61E-2)	(3.16E+4, 6.11E-2)	(1.00E+6, 4.90E-3)
`image`	(4.22E+4, 1.22E-3)	(2.37E+4, 1.10E-2)	(4.22E+4, 1.22E-3)	(1.00E+2, 2.00E-3)
`satimage`	(3.16E+1, 8.93E-2)	(3.16E+1, 8.93E-2)	(3.16E+1, 8.93E-2)	(3.16E+0, 1.00E-1)
`pendigits`	(1.36E+1, 1.14E-1)	(2.37E+3, 1.56E-1)	(3.16E+1, 6.08E-2)	(3.16E+0, 7.12E-2)

4.2. The Purification Ability of OOR

This subsection report the results of two voting analyses for the previous prediction results of OOR and one-vs.-one on the `image` testing dataset. Results show that OOR has a strong purification ability to filter out nuisance votes.

In the first voting analysis, for each pair (actual class c_i, predictive class c_j), calculate the average vote value ζ obtained from the subclassifiers that classify the c_i-class testing points into c_j-class, where $i, j = 1, 2, \ldots, k$ and $0 \leq \zeta \leq k-1$. In Table 4, a value in cell (c_2, c_4) belonging to the one-vs.-one scheme is 4.95. This means that through the $(k-1)$ c_4-related subclassifiers of one-vs.-one, the predictive class c_4 gains 4.95 average votes among the testing points, which actually belong to class c_2. In other words, once the majority

Table 3: Accuracy reports on nine UCI datasets

datasets	TSSVM/OOR	one-vs.-one	one-vs.-rest	LIBSVM
iris	**98.00**	97.33	97.33	96.33 (97.33)
tae	**70.30**	66.60	67.58	62.00
wine	**99.44**	98.23	98.33	98.33 (99.44)
dna*	**95.36**	93.51	92.24	95.20 (95.45)
glass	**75.29**	72.14	72.59	71.52 (71.50)
segment*	97.10	96.10	96.54	**97.15** (97.40)
image	**92.05**	90.48	90.86	90.62
satimage*	**91.55**	90.75	91.05	91.28 (91.30)
pendigits*	97.71	97.17	97.68	**98.23**

Notation * indicates the use of the reduced kernel technique on `dna`, `segment`, `satimage` and `pendigits`, the corresponding reduced sets are randomly selected by 15%, 10%, 15% and 5% respectively. This technique is used only for TSSVM/OOR and one-vs.-rest. All data problems are solved by one-vs.-one and LIBSVM with full-kernel. The available results of LIBSVM with grid-search (15×15) in [17] are also announced in parentheses. The bold number means the highest accuracy rate among these multi-class SVMs.

strategy is applied, the second highest average value 4.95 easily leads the one-vs.-one scheme to make wrong estimations. However, the corresponding second high value reported from the OOR scheme in cell (c_2, c_4) decreases dramatically to 0.02. The OOR scheme not only keeps the average value 6.00 in an idea correct estimation from the cell (c_2, c_2) of one-vs.-one table, but it also succeeds in removing those wrong estimations in $(c_2, c_{j=\{1,...,7\}\setminus\{2\}})$ to (c_2, c_r), where c_r means the rest-class embedded in each ternary subclassifier of OOR. The average value of (c_i, c_r) is lower than $k(k-1)/2$. Results show that the OOR scheme is effective in nuisance-vote filtering and successful in pacifying the tense voting results from the one-vs.-one scheme.

For the second voting analysis, we look into the voting behavior of OOR and one-vs.-one for the testing points. Many testing points are misclassified by the one-vs.-one scheme, but are predicted correctly by the OOR scheme. Table 5 only reports results from partial testing points. We list these examples according to the original order from UCI. The function $c_j(x)$ records the number of times that $k(k-1)/2$ subclassifiers classify the data point x into predictive class c_j, where $c_j(x) \leq k-1$ for $j \in \{1, ..., 7\}$ and the rest-class $c_r(x) \leq k(k-1)/2$. Note that when predictive classes have the same number

Table 4: Average voting analysis on `image` testing dataset

one-vs.-one									
Actual class	Predictive class								Sum of per row
	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_r	
c_1	5.78	0	3.85	3.80	4.56	1.60	1.40	-	21
c_2	1.00	6.00	3.00	4.95	2.05	4.00	0	-	21
c_3	3.53	0.10	5.77	3.57	4.84	1.80	1.38	-	21
c_4	2.51	1.15	3.17	5.79	3.75	4.14	0.49	-	21
c_5	3.77	0	4.06	4.22	5.66	1.74	1.55	-	21
c_6	2.50	0.51	3.56	4.98	2.94	6.00	0.51	-	21
c_7	2.00	0	4.37	3.38	3.90	1.39	5.96	-	21

OOR									
Actual class	Predictive class								Sum of per row
	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_r	
c_1	5.73	0	0	0.44	0.03	0	0	14.79	21
c_2	0	6.00	0	0.02	0	0	0	14.98	21
c_3	0.04	0	5.29	0.11	1.22	0.01	0	14.33	21
c_4	0.03	0.04	0.15	4.30	0.17	0.30	0	16.01	21
c_5	0.05	0	0.91	0.70	3.94	0.01	0	15.40	21
c_6	0	0	0.01	0.04	0.01	6.00	0	14.94	21
c_7	0	0	0.02	0	0	0.06	5.95	14.96	21

c_r means average sum of votes for the rest-class used only in the OOR scheme. (The ideal cases in OOR's table are $(c_i, c_i) = k-1$ and $(c_i, c_r) = k(k-1)/2 - (k-1)$)

of votes, x can be randomly assigned to one class among these classes. The OOR scheme succeeds in avoiding the nuisance votes generated from one-vs.-one, dramatically converting the wrong estimations into correct ones based on the majority strategy.

Both the voting analyses above show that OOR has the strong purification ability, which is the main reason why OOR can effectively increase the prediction accuracy under the more reasonable voting scheme. The following subsection uses detailed error analyses to show that not only the purification ability of OOR can raise the prediction accuracy, but it can effectively pacify the tense and unstable phenomenon of one-vs.-one to achieve more confidential prediction results.

4.3. The Prediction Confidence of OOR

This subsection attempts to identify the confidence-level in which most testing points are determined by one-vs.-one and OOR, and provide two kinds of detailed error analyses to measure each confidence-level. The rank of a confidence-level with two error analyses served as the criteria for evaluating

Table 5: Voting behavior of OOR and one-vs.-one on `image` testing subset

Index of x	Class label	OOR								one-vs.-one						
		c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_r	c_1	c_2	c_3	c_4	c_5	c_6	c_7
1082	1	**5**	0	0	1	0	0	0	15	4	0	5	**6**	3	2	1
1084	1	**4**	0	0	3	0	0	0	14	4	0	5	**6**	3	2	1
1089	1	**5**	0	0	4	0	0	0	12	4	0	**5**	**5**	4	2	1
1100	1	**6**	0	0	4	0	0	0	11	5	0	4	**6**	3	2	1
1112	1	**5**	0	0	1	0	0	0	15	4	0	**6**	4	4	2	1
1122	1	**6**	0	0	2	0	0	0	13	5	0	4	**6**	3	2	1
1131	1	**6**	0	0	1	0	0	0	14	5	0	3	**6**	4	2	1
1152	1	**5**	0	0	2	0	0	0	14	**5**	0	**5**	4	4	2	1
1153	1	**6**	0	0	2	0	0	0	13	5	0	4	**6**	3	2	1
420	3	0	0	**6**	0	4	0	0	11	3	0	**5**	**5**	**5**	2	1
422	3	0	0	**5**	2	0	0	0	14	3	0	5	**6**	4	2	1
428	3	0	0	**6**	0	4	0	0	11	3	0	**5**	**5**	**5**	2	1
443	3	0	0	**6**	0	0	0	0	15	2	1	4	**6**	3	5	0
452	3	0	0	**6**	0	0	0	0	15	4	0	5	3	**6**	1	2
485	3	0	0	**6**	2	0	0	0	13	3	0	5	**6**	4	2	1
1249	3	0	0	**5**	0	0	0	0	16	3	0	5	**6**	4	2	1
1255	3	0	0	**6**	0	5	0	0	10	3	0	**5**	**5**	**5**	2	1
1261	3	0	0	**6**	0	5	0	0	10	3	0	**5**	**5**	**5**	2	1
239	4	0	0	0	**5**	0	0	0	16	2	1	3	5	**6**	4	0
251	4	0	0	1	**5**	0	0	0	15	3	0	**6**	4	5	2	1
266	4	0	0	0	**5**	0	0	0	16	2	1	3	5	**6**	4	0
272	4	0	0	0	**5**	0	0	0	16	4	1	2	5	**6**	3	0
284	4	0	0	0	**5**	0	0	0	16	4	0	**5**	4	**5**	2	1
290	4	0	0	0	**6**	0	0	0	15	2	1	3	5	**6**	4	0
299	4	0	0	0	**5**	**5**	0	0	11	3	0	5	4	**6**	2	1
311	4	0	0	0	**5**	0	0	0	16	3	1	2	5	**6**	4	0
321	4	0	0	0	**6**	4	0	0	11	3	1	3	5	**6**	3	0
954	5	0	0	0	1	**6**	0	0	14	4	0	3	**6**	5	2	1
970	5	0	0	0	0	**2**	0	0	19	**5**	0	3	**5**	**5**	2	1
981	5	0	0	0	0	**6**	0	0	15	**5**	0	4	4	**5**	2	1
1013	5	0	0	1	0	**4**	0	0	16	3	0	**6**	4	5	2	1
1397	5	0	0	0	0	**2**	0	0	19	4	0	**5**	4	**5**	2	1
1731	5	0	0	0	3	**4**	0	0	14	2	0	3	**6**	5	4	1
1734	5	0	0	1	0	**5**	0	0	15	3	0	4	**6**	5	2	1
2076	5	0	0	0	0	**2**	0	0	19	**5**	0	**5**	3	**5**	1	2
2084	5	0	0	**5**	0	**5**	0	0	11	3	0	**6**	4	5	2	1

This table only reports a subset for those testing instances which are misclassified by one-vs.-one but they are predicted correctly by OOR. c_r means average sum of votes for the rest-class used only in the OOR scheme. The bold number represents the highest votes among 7 classes.

prediction confidences of these two schemes. Take the previous prediction results of OOR and one-vs.-one on the data problems of `image`, `satimage`, and `pendigits`, which have provided the testing datasets with $k > 3$, as examples. For a testing point x, we are concerned only with the highest two values of $V_1(x)$ and $V_2(x)$ among $c_1(x), c_2(x), \ldots, c_k(x)$, and define the confidence-level by $\eta(x)$, where $0 \le \eta(x) = V_1(x) - V_2(x) \le k - 1$. These two detailed error analyses are called *conditional confidence error rate* (c.c.e.r.) and *conditional misclassified error rate* (c.m.e.r.), defined as follows:

$$\begin{aligned}(c.c.e.r.)_\eta &= \frac{Number\ of\ misclassified\ points\ in\ level\ \eta}{Number\ of\ points\ in\ level\ \eta} \\ (c.m.e.r.)_\eta &= \frac{Number\ of\ misclassified\ points\ in\ level\ \eta}{Number\ of\ total\ misclassified\ points}\end{aligned} \tag{18}$$

For the one-vs.-one scheme on the `image` testing dataset, Table 6 shows that there are 1715 testing points and the 200 (#errs) misclassified points voted under confidence-level 1. The c.c.e.r. is 7.11% and c.m.e.r. is 61%. In other words, most testing points are determined by one-vs.-one under the lower confidence-level 1 with high conditional error rates. On the other hand, most testing points are not only determined by OOR under the highest confidence-level 6, but the 2.72% c.c.e.r. and 25.75% c.m.e.r. (#errs=167) results are comparatively smaller. the numerical results in Tables 7 and 8 show the same phenomenon on the prediction analyses of one-vs.-one and OOR.

Table 6: Confidence-level error analyses on 2100 image testing points

Scheme	$\eta(x)$	0	1	2	3	4	5	6
one-vs.-one	#points	79	**1715**	306	0	0	0	0
	#errs	62	122	16	0	0	0	0
	c.c.e.r. (%)	78.48	7.11	5.23	-	-	-	-
	c.m.e.r. (%)	31.00	61.00	8.00	-	-	-	-
OOR	#points	39	94	122	65	56	144	**1580**
	#errs	21	32	26	18	10	17	43
	c.c.e.r. (%)	53.85	34.04	21.31	27.69	17.86	11.81	2.72
	c.m.e.r. (%)	12.57	19.16	15.57	10.78	5.99	10.18	25.75

The bold number denotes that most testing points are determined under the confidence-level.

Table 7: Confidence-level error analyses on 2000 `satimage` testing points

Scheme	$\eta(x)$	0	1	2	3	4	5
one-vs.-one	#points	37	**1953**	10	0	0	0
	#errs	8	171	6	0	0	0
	c.c.e.r. (%)	21.62	8.76	60.00	-	-	-
	c.m.e.r. (%)	4.32	92.43	3.24	-	-	-
OOR	#points	15	35	49	24	32	**1845**
	#errs	6	16	16	5	15	111
	c.c.e.r. (%)	40	45.71	32.65	20.83	46.88	6.02
	c.m.e.r. (%)	3.55	9.47	9.47	2.96	8.88	65.7

The bold number denotes that most testing points are determined under the confidence-level.

Table 8: Confidence-level error analyses on 3498 `pendigits` testing points

Scheme	$\eta(x)$	0	1	2	3	4	5	6	7	8	9
one-vs.-one	#points	7	**3188**	303	0	0	0	0	0	0	0
	#errs	4	83	12	0	0	0	0	0	0	0
	c.c.e.r. (%)	57.14	2.60	3.96	-	-	-	-	-	-	-
	c.m.e.r. (%)	4.04	83.84	12.12	-	-	-	-	-	-	-
OOR	#points	4	26	29	13	10	4	7	13	20	**3372**
	#errs	4	14	16	3	3	3	2	2	7	26
	c.c.e.r. (%)	100	53.85	55.17	23.08	30	75	28.57	15.38	35.00	0.77
	c.m.e.r. (%)	5.00	17.50	20.00	3.75	3.75	3.75	2.50	2.50	8.75	32.50

The bold number denotes that most testing points are determined under the confidence-level.

These experiments indicate that a multi-class classification task is usually done by one-vs.-one under confidence-level 1, which is a quite tense and unstable situation because $V_2(x)$ can easily overcome $V_1(x)$, decreasing prediction confidence. However, the OOR scheme without bothering nuisance votes successfully pacifies the circumstance, achieve high confidence. Thus, the accuracy rate reported from OOR will be more convincing even if these two accuracy rates are the same.

4.4. The Detection Ability of OOR for Hidden Classes

To demonstrate the detection ability of the proposed OOR method for hidden classes, this study includes a "leave-one-class-out" experiment on the pendigits dataset. In this case, the training phase uses 9 classes of data points, while the testing phase involves 10 classes. In our hidden-class detection criterion, an instance is a hidden class if all given classes gain fewer than 8 votes. The detection results of OOR will be compared with the results of one-vs.-one and one-vs.-rest strategies. We report the testing accuracy as well as the detection rate for hidden class for all strategies. For one-vs.-one, the detection criterion is the same with OOR. The one-vs.-rest scheme uses the criterion that all output values of the decision functions are negative. The results show that OOR significantly outperforms the one-vs.-one and one-vs.-rest schemes in the hidden class detection rate. In the testing accuracy, OOR is slightly better than others because the hidden class instances represent only a small portion in the entire testing set.

Table 9: The detection results for hidden classes on pendigits dataset

Pendigits	Overall Accuracy Rate (%)			Hidden-Class Detection Rate (%)		
Scheme	OOR	one-vs.-one	one-vs.-rest	OOR	one-vs.-one	one-vs.-rest
Hidden (c_0)	**89.88**	87.51	86.88	**62.81**	12.95	5.51
Hidden (c_1)	**90.62**	87.99	88.82	**53.30**	16.48	26.10
Hidden (c_2)	**89.34**	86.45	88.22	**54.40**	2.75	20.33
Hidden (c_3)	**92.42**	88.42	91.40	**75.00**	14.88	46.73
Hidden (c_4)	94.11	89.71	**94.20**	**98.35**	21.15	74.18
Hidden (c_5)	**89.54**	88.65	88.62	**45.07**	18.81	18.51
Hidden (c_6)	**94.54**	89.91	90.39	**96.43**	25.60	36.01
Hidden (c_7)	**95.17**	88.71	94.68	**89.29**	18.41	67.58
Hidden (c_8)	**92.71**	88.59	91.71	**69.94**	25.30	50.30
Hidden (c_9)	**90.82**	89.28	88.65	**55.65**	21.13	17.56

The bold number means the highest accuracy rate among these methods.

5. Conclusions

This study proposes a smooth support vector machine for multi-class classification problems based on the single machine approach. The proposed MSSVM is an unconstrained smooth and convex minimization problem. Due to these features, MSSVM can be solved by a Newton-Armijo algorithm, which is globally convergent with quadratic time. This study implements

MSSVM for ternary classification problems, labeling it as TSSVM. For the case of a class number greater than three, the proposed one-vs.-one-vs.-rest (OOR) scheme decomposes the problem into a series of ternary classification problems. The key difference between this OOR scheme and the one-vs.-one scheme is that the OOR scheme allows us to classify a new unseen label instance to the *rest* class. Thus, the OOR scheme can filter out many nuisance votes, offering a more reasonable way to solve multi-class classification problems. The numerical results and detailed error analysis in this study show that the proposed method achieves higher classification confidence. The proposed OOR scheme also has the ability to detect hidden-class instances in solving multi-class classification problems.

References

[1] E.L. Allwein, R.E. Schapire, Y. Singer, Reducing multiclass to binary: a unifying approach for margin classifiers, Journal of Machine Learning Research 1 (2000) 113-141.

[2] A. Asuncion, D.J. Newman, UCI machine learning repository, Irvine, CA: University of California, School of Information and Computer Science, 2007 <http://www.ics.uci.edu/∼mlearn/MLRepository.html>.

[3] L. Bottou, C. Cortes, J. Denker, H. Drucker, I. Guyon, L. Jackel, Y. LeCun, U. Muller, E. Sackinger, P. Simard, V. Vapnik, Comparison of classifier methods: a case study in handwriting digit recognition. in: International Conference on Pattern Recognition, IEEE Computer Society Press, 1994, pp. 77-87.

[4] E. Bredensteiner, K.P. Bennett, Multicategory classification by support vector machines, Computational Optimization and Applications 12 (1999) 53-79.

[5] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (1998) 121-167.

[6] C.C. Chang, C.J. Lin, LIBSVM: A Library for Support Vector Machines, 2001, Software available at <http://www.csie.ntu.edu.tw/ cjlin/libsvm>.

[7] F.H. Clarke, Optimization and Nonsmooth Analysis, SIAM, Philadelphia, PA, 1990.

[8] C. Cortes, V. Vapnik, Support vector networks, Machine Learning 20 (1995) 273-297.

[9] K. Crammer, Y. Singer, Improved output coding for classification using continuous relaxation, in: Proc. of the Thirteenth Annual Conference on Neural Information Processing Systems, 2000.

[10] K. Crammer, Y. Singer, On the learnability and design of output codes for multiclass problems, Machine Learning 47 (2002) 201-233.

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

[12] T.G. Dietterich, B. Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artificial Intelligence Research 2 (1995) 263-286.

[13] K.T. Fang, , Y. Wang, Number-Theoretic Methods in Statistics. Chapmman & Hall, London, 1994.

[14] D.S. Felsenthal, M. Machover, Ternary voting games, International Journal of Game Theory 26 (1997) 335-351.

[15] J. Friedman, Another approach to polychotomous classification Technical Report, Department of Statistics, Stanford University, 1996.

[16] J. Fürnkranz, Round robin classification, Journal of Machine Learning Research (2002) 721-747.

[17] C.W. Hsu, C.J. Lin, A comparison on methods for multi-class support vector machines, IEEE Transactions on Neural Networks 13 (2002) 415-425.

[18] C.M. Huang, Y.J. Lee, D.K.J. Lin, S.Y. Huang, Model selection for support vector machines via uniform design, A special issue on Machine Learning and Robust Data Mining of Computational Statistics and Data Analysis 52 (2007) 335-346.

[19] E. Hüllermeier, S. Vanderlooy, Combing predictions in pairwise classification: An optimal adaptive voting strategy and its relation to weighted voting, Pattern Recognition 43 (2010) 128-142.

[20] H.C. Kim, S. Pang, H.M. Je, D. Kim, S.Y. Bang, Constructing support vector machine ensemble, Pattern Recognition 36 (2003) 2757-2767.

[21] S. Knerr, L. Personnaz, G. Dreyfus, Single-layer learning revisited: a stepwise procedure for building and training a neural network, in: J. Fogelman (Ed.), Neurocomputing: Algorithms, Architectures and Applications, Springer-Verlag, 1990.

[22] U. Kreβel, Pairwise classification and support vector machines, in: B. Schölkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel Methods - Support Vector Learning, MIT Press, Cambridge, MA, 1999, pp. 255–268.

[23] Y. Lee, Y. Lin, G. Wahba, Multicategory support vector machines, in: Proc. of the 33rd Symposium on the Interface, 2001.

[24] Y.J. Lee, S.Y. Huang, Reduced support vector machines: a statistical theory, IEEE Transactions on Neural Networks 18 (2007) 1-13.

[25] Y.J. Lee, O.L Mangasarian, SSVM: A smooth support vector machine. Computational Optimization and Applications 20 (2001) 5-22.

[26] Y.J. Lee, O.L Mangasarian, RSVM: Reduced support vector machines. in: First SIAM International Conference on Data Mining, Chicago, 2001.

[27] O.L. Mangasarian, Generalized support vector machines, in: A. Smola, P. Bartlett, B. Schölkopf, D. Schuurmans (Eds.), Advances in Large Margin Classiers, Cambridge, MA, MIT Press, 2000, pp. 135-146.

[28] MATLAB, Users Guide, The MathWorks, Inc., Natick, MA 01760, 1994-2006. <http://www.mathworks.com>.

[29] D. Michie, D.J. Spiegelhalter, C.C. Taylor, Machine Learning, Neural and Statistical Classification, Prentice-Hall, Englewood Cliffs, N.J., 1994.

[30] J.C. Platt, N. Cristianini, J. Shawe-Taylor, Large margin DAGs for multiclass classification, in: Advances in Neural Information Processing Systems, volume 12, MIT Press, 2000, pp. 547-553.

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

[32] A. Smola, B. Schölkop, Sparse greedy matrix approximation for machine learning, in: Proc. of 17th International Conference on Machine Learning, San Francisco, CA, 2000, pp. 911-918.

[33] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.

[34] J. Weston, C. Watkins, Multi-class support vector machines, in: M. Verleysen (Ed.), Proc. of ESANN99, Brussels D. Facto Press, 1999, pp. 219- 224.

[35] Y.-C. F. Wang, D. Casasent, New support vector-based design method for binary hierarchical classifiers for multi-class classification problems, Neural Networks 21 (2008) 502-510.

[36] C.K.I Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds), Advances in Neural Information Processing Systems 13, MIT Press, Cambridge, MA, USA, 2001, pp. 682-688.