Towards Optimal Discriminating Order for Multiclass Classification

Dong Liu*†, Shuicheng Yan†, Yadong Mu†, Xian-Sheng Hua‡, Shih-Fu Chang§ and Hong-Jiang Zhang¶
*School of Computer Science and Technology, Harbin Institute of Technology, Harbin, P. R. China
†Department of Electrical and Computer Engineering, National University of Singapore, Singapore
‡Microsoft Research Asia, Beijing, P. R. China
§Department of Electrical Engineering, Columbia University, New York, USA
¶Microsoft Advanced Technology Center, Beijing, P. R. China
Email: dongliu.hit@gmail.com, {eleyans,elemuy}@nus.edu.sg, {xshua,hjzhang}@microsoft.com, sfchang@ee.columbia.edu

***Abstract*—In this paper, we investigate how to design an optimized discriminating order for boosting multiclass classification. The main idea is to optimize a binary tree architecture, referred to as Sequential Discriminating Tree (SDT), that performs the multiclass classification through a hierarchical sequence of coarse-to-fine binary classifiers. To infer such a tree architecture, we employ the constrained large margin clustering procedure which enforces samples belonging to the same class to locate at the same side of the hyperplane while maximizing the margin between these two partitioned class subsets. The proposed SDT algorithm has a theoretic error bound which is shown experimentally to effectively guarantee the generalization performance. Experiment results indicate that SDT clearly beats the state-of-the-art multiclass classification algorithms.**

***Keywords*-Discriminating Order, Multiclass, Sequential Discriminating Tree.**

I. INTRODUCTION

Supervised multiclass learning aims at deriving a function that can accurately assign class labels to instances where the label set is of finite cardinality yet contains at least three elements. Such a learning problem is becoming increasingly important in various disciplines including natural language processing [1], computer vision [2] and computational biology [3].

$\ell(x) = (1 - x)_+$

Many algorithms have been proposed to solve the multiclass classification problem, varying both on the decomposition of the multiclass problem into a set of binary problems and on the "all-together" single formulation used to optimize multiclass discrimination. However, these algorithms suffer from such limitations as the high variance in the output prediction and the computational burden in solving a large scale optimization problem (see Section II for details). Therefore, how to effectively solve multiclass classification is still an unsolved research problem.

The multiclass classification problem needs to discriminate samples from $N(N > 2)$ classes, thus intuitively the procedure is inevitably implemented in a stepwise elimination manner, where a subset of the N classes are discriminated at first, followed by the further discrimination of the remaining classes. For such stepwise elimination process, an appropriate discriminating order is critical, especially when using linear classifiers. As an example, Figure 1(a) depicts a case where the discriminating order has a direct impact on the final decision of multiclass discrimination. As can be observed, the 4-class data are linearly inseparable under "one-versus-all" settings [4], yet become linearly separable under the deliberate discriminating order as presented in the figure. However, existing multiclass classification algorithms [5], [6], [7], [8], [9], [10], [11] typically do not take the discriminating order into consideration, which directly motivates our work in this paper.

In this work, we propose a new algorithm towards the optimal discriminating order in multiclass discriminating procedure. The proposed algorithm, referred to as Sequential Discriminating Tree (SDT), performs a coarse-to-fine multiclass discrimination by recursively dividing the classes into two subsets until all classes are completely separated. As shown in Figure 1(b), the top-down architecture of SDT renders an optimized discriminating order with the most separable class clusters discriminated at the root node, more sophisticated classes discriminated at the lower layers and so on until the decisive leaf nodes. We employ the constrained large margin clustering procedure to discover such a tree architecture and perform binary discrimination in the sequence, by identifying the maximum margin between two class clusters while enforcing the samples in the same class to locate at the same side of the hyperplane. In addition, the theoretic analysis indicates that such a learning process is guaranteed by an generalization error bound. Extensive experiments well validate the superiority of the proposed SDT algorithm over those state-of-the-art multiclass classification algorithms.

The rest of this paper is organized as follows. In Section II, we briefly introduce the related work on the state-of-the-art multiclass classification algorithms. Then we propose the sequential discriminating tree algorithm in Section III. Section IV provides the algorithmic theoretic analysis including time complexity and generalization error bound. In Section V, we report the experimental results on a board range of data sets. The last section concludes this work along with future work discussion.

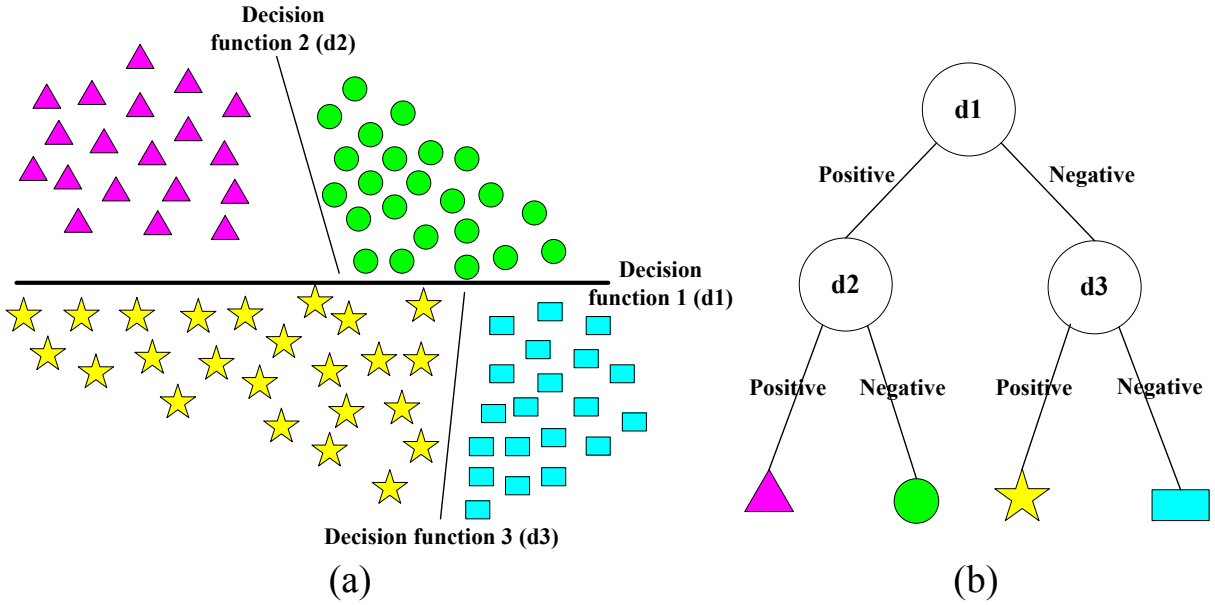


Figure 1. (a) The 4-class data cannot be well separated with the simple linear classifiers unless using the deliberate discriminating order presented in the figure. (b) The architecture of SDT for the given task that sequentially performs the discriminating in an optimal order. Note that although only linear planes are shown here for illustration, we will present extensions to nonlinear kernels later in the paper.

II. Related Work

Many multiclass classification algorithms [5], [6], [7], [8], [9], [10] have been developed during the last decade. As this work focuses on solving multiclass classification problem based on large margin optimization, Here we only briefly review the multiclass learning algorithms that are closely related to large margin learning.

There are four popular large margin N-class classification algorithms in literature. The first one is to construct and combine N binary Support Vector Machines (SVM) [4]. The i-th SVM is trained with all of the training samples in the i-th class as positive samples, and all other training samples in the other $N-1$ classes as negative samples. Therefore, the training time of this approach scales linearly with the number of classes. We refer to the classifier trained in this way as one-versus-all SVM (OVA SVM). In the testing phase, the testing sample is evaluated against the N binary SVMs, and the one which gives the highest decision value is chosen as the predicted class. However, the unbalanced numbers of the positive and negative samples in the binary SVM training cause issue with the "OVA" strategy, especially when it has few training samples per class. Besides, there is no theoretic bound for the generalization error of the OVA SVM.

The second algorithm is called one-versus-one SVM (OVO SVM) [12]. OVO SVM trains a binary SVM classifier for any two out of the N classes, which consequently results in $N(N-1)/2$ binary SVM classifiers. To combine these classifiers, Kreßel [5] suggested a Max Win strategy in which each binary classifier casts one vote for its preferred class, and the final result is the class with the largest vote. To implement the above strategy, each test sample has to be presented to a number of $N(N-1)/2$ classifiers, which thus results in slower testing, especially when the number of the classes is large. To enforce fast testing for OVO SVM, Platt [6] proposed the DAGSVM algorithm, the training of which is the same as that for OVO SVM. In the testing phase, the algorithm utilizes a rooted binary directed acyclic graph to make a decision, which results in only $N-1$ binary evaluations. The disadvantage of this approach, however, is that the individual binary SVM classifiers tend to overfit the concerned 2-class training samples, hence the learned discriminating boundaries cannot provide meaningful generalization analysis for the unseen testing samples from the other $N-2$ classes.

The third algorithm is to construct a piecewise separation of the N classes in an all-together optimization formulation [8], [10], [13]. The basic idea is to build N binary classification rules where the i-th function separates training samples of the i-th class from the other training samples. Hence there are N decision functions but all are learned by solving a single optimization problem. For any testing sample, the class with the largest decision value is output as the decision. Therefore, the entire scenario of this algorithm is still similar to OVA SVM.

The fourth algorithm for multiclass classification arranges the classes into a hierarchial binary tree such that the involved classes at each internal node are divided into two clusters, one for each child node [9], [14]. The division process continues until the leaf node contains only a single

class. At each node of the tree, a binary SVM classifier is constructed to make discrimination between the two child class clusters. Although this approach also relies on organizing a number of binary classifiers in a tree architecture, it separates the tree construction from the classifier learning and neglects the data labels in determining the discriminating order. Therefore, the entire scenario of this algorithm is essentially different from our proposed SDT algorithm which directly optimizes ordinal sequence of binary divisions in the tree construction process by taking the class labels into consideration.

There are also some pieces of works [15], [16] that address the multiclass problems by simply ensembling multiple candidate hierarchial trees of a given problem with C4.5 and logistic regression as base learners. Besides, the work on Error-Correcting Output Codes [17], [18] allows a combination of classifiers which benefit from error-correcting principles. The work in [19] provides a tree decomposition framework for large-scale SVM learning. Despite the above progresses, the main focus of these works is the decision tree ensemble, which, however, is essentially different from determining the optimal discriminating order exploited in this paper. On the other hand, our proposed algorithm can also be utilized as a pre-processing of these ensemble algorithms, which may further improve the performance.

III. Sequential Discriminating Tree

As aforementioned, the discriminating order has a great impact on the performance of multiclass classification. However, identifying the optimal discriminating order and incorporating it into the multiclass learning process is a challenging task. The most straightforward approach to tackle this issue is to rank the classes according to the their difficulties in discrimination, but it is obviously not the best solution since the correlation among the classes are not exploited. Another alternative is to estimate the merits of different tentative discriminating orders of the classes, however, this may result in the combinatorial explosion, especially when the class number is large.

To address this problem, we propose a Sequential Discriminating Tree (SDT) algorithm which derives the optimal discriminating order through a hierarchical binary partitioning of the classes. It is worth noting that such a partition based learning strategy is critical in identifying the optimal discriminating order in multiclass classification. Taking the classification problem in Figure 1 as an example, if we perform the class-wise discrimination, the data cannot be separated with the simple linear classifiers, whereas if we apply the partition based discriminating, the problem becomes linearly separable. On the other hand, the binary partition of the classes breaks the whole difficult problem into its subparts, making each of them easier to be solved.

Besides explicitly exploiting the separability among the classes, the proposed SDT algorithm uses a binary tree architecture to represent the discriminating order of multiclass classification, where the root node denotes the first discriminating function and each leaf node denotes the final decision of one specific class. Starting from the N-class training samples, the SDT sequentially partitions the data such that the samples in the same class are grouped into the same subset until every subset consists of samples from a single class. By doing so, we convert the combinational optimization problem into a sequence of computationally practical binary partition problems, making the proposed SDT algorithm more applicable in real world problems. In this section, we will first introduce the learning strategy for the SDT induction and then describe how SDT makes decision for the testing samples based on the learned tree architecture.

A. Tree Induction

The induction of SDT focuses on recursively partitioning the current training samples into two non-overlapping subsets by a binary discriminating function, which is applied as the node classifier of SDT. Therefore, the key ingredient in SDT induction concerns how to perform an effective binary partition at each non-leaf node. To this end, two important characteristics are desired: (1) the training samples belonging to the same class should be grouped together in the same subset, and (2) to ensure the generalization ability to the unseen testing samples, the partition function should have the largest possible margin. These two criteria can be well realized via the constrained large margin binary clustering algorithm which was first introduced in [20]. The authors in [21] further proposed a constrained large margin clustering algorithm and applied it into the task of semi-supervised hashing. In this work, we employ this constrained clustering algorithm as the binary partition procedure at each node of SDT. For completeness, we will first elaborate on the algorithm in [21].

Given a collection of samples $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$, where each $\mathbf{x}_i$ denotes the feature vector of a sample and n denotes the total number of samples in the collection. The constrained large margin clustering algorithm aims at finding an optimal binary partition hyperplane vector $f(\mathbf{x}_i) = \omega^\top \mathbf{x}_i + b$, as well as the corresponding binary partition label y_i[1], where ω denotes the hyperplane function and b is the offset scalar. In the binary partition procedure of the constrained large margin clustering algorithm, samples of the same class should be partitioned into the same side of the hyperplane, and thus trigger a penalty when they are projected into different sides of the hyperplane. Denote the constraint set for all the training samples as Θ_s, in which $(i, j) \in \Theta_s$ denotes that the training samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are from the

[1]Here y_i does not denote the class label of the sample $\mathbf{x}_i$, instead, it is used to represent which side of the hyperplane the sample $\mathbf{x}_i$ locates at, where $y_i = +1$ indicates that $\mathbf{x}_i$ is at the positive side while $y_i = -1$ shows that $\mathbf{x}_i$ is at the negative side.

same class. Based on these constraints, the objective of the large margin binary clustering can be formulated as

$$\mathcal{J}_\omega = \Omega(\omega) + \lambda_1 \sum_i \ell\left(-y_i f(\mathbf{x}_i)\right) + \lambda_2 \sum_{(i,j)\in\Theta_s} \hbar((i,j)), \quad (1)$$

where $\Omega(\omega) = \frac{1}{2}\|\omega\|^2$ denotes the regularization term, and $\ell(\cdot)$ is the hinge loss function defined as $\ell(y_i f(\mathbf{x}_i)) = (1 - y_i f(\mathbf{x}_i))_+$. Moreover, the constraint loss term $\hbar(\cdot)$ can be defined as:

$$\hbar((i,j)) = \begin{cases} 0, & y_i = y_j, \\ (-y_i y_j)_+, & y_i \neq y_j. \end{cases} \quad (2)$$

Following the notations in the max-margin formulation [22], the objective function in Eq. (1) can be written as:

$$\begin{aligned} \min_{\omega,b,\xi,\zeta,y} \quad & \frac{1}{2}\|w\|^2 + \frac{\lambda_1}{n}\sum_i \xi_i + \frac{\lambda_2}{n}\sum_{(i,j)\in\Theta_s} \zeta_{ij} \quad (3) \\ s.t. \quad & y_i(\omega^T\mathbf{x}_i + b) + \xi_i \geq 1,\ \xi_i \geq 0,\ \forall\ i, \\ & y_i y_j + \zeta_{ij} \geq 0,\ \zeta_{ij} \geq 0,\ \forall (i,j) \in \Theta_s. \end{aligned}$$

Obviously, the above constrained large margin binary clustering objective is only designed for the raw feature vectors. To handle the non-linear relations among the samples, the kernel trick can be applied, where a mapping function $\phi(\cdot)$ is employed to transform the features into the Reproducing Kernel Hilbert Space (RKHS). In this case, according to the representer theorem [23], the vector ω in Eq. (1) can be written as a linear combination of the mapped feature vectors, i.e., $\omega = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i)$, where α_i represents the weighting parameter. Moreover, by definition, y_i equals to $\text{sign}(\omega^T\mathbf{x}_i + b)$ [4], then $y_i(\omega^T\mathbf{x}_i + b)$ is consequently non-negative and can be equivalently expressed as $|\omega^T\mathbf{x}_i + b|$. Furthermore, $y_i y_j$ can be replaced with $(\omega^T\mathbf{x}_i + b)(\omega^T\mathbf{x}_j + b)$, such that the variables y_i's are eliminated in the optimization. Finally, by incorporating $\omega = \sum_{i=1}^n \alpha_i \phi(\mathbf{x}_i)$ into the optimization objective, we get the relaxed formulation:

$$\begin{aligned} \min_{\alpha,b,\xi,\zeta} \quad & \frac{1}{2}\alpha^T G\alpha + \frac{\lambda_1}{n}\sum_i \xi_i + \frac{\lambda_2}{n}\sum_{(i,j)\in\Theta_s} \zeta_{ij} \quad (4) \\ s.t. \quad & |\alpha^T k_i + b| + \xi_i \geq 1,\ \forall\ i, \quad (5) \\ & (\alpha^T k_i + b)(\alpha^T k_j + b) + \zeta_{ij} \geq 0, \quad (6) \\ & \xi_i \geq 0,\ \forall\ i, \\ & \zeta_{ij} \geq 0,\ \forall\ (i,j) \in \Theta_s, \end{aligned}$$

where G is an $n \times n$ Gram matrix computed from the n training samples, and k_i denotes the inner products between the i-th training sample and all n samples.

The objective in Eq. (4) is convex, and the constraints in Eqs. (5) (6) can all be expressed as the difference of two convex functions after simple algebra derivation. Therefore, the *Constrained Concave-Convex procedure* (CCCP) [24], [25], can be used to solve this optimization problem. However, $|\alpha^T k_i + b|$ in Eq. (5) is non-smooth, and thus its gradient possibly does not exist at some locations. To make CCCP applicable, we can replace their gradients by subgradients at the t-th iteration of CCCP as follows:

$$\begin{aligned} \partial_\alpha |\alpha^T k_i + b| &= k_i \cdot \text{sign}(\alpha_t^T k_i + b_t), \\ \partial_b |\alpha^T k_i + b| &= \text{sign}(\alpha_t^T k_i + b_t), \end{aligned}$$

where α_t denotes the solution of the weighting vector α at the t-th CCCP iteration.

Based on the above subgradients, the Taylor approximations for the terms in constraints Eqs. (5) (6) can be written as:

$$|\alpha^T k_i + b| \approx (\alpha^T k_i + b) \cdot \text{sign}(\alpha_t^T k_i + b_t), \quad (7)$$

$$\frac{1}{2}\widetilde{\alpha}^T M_{ij}\widetilde{\alpha} \approx \widetilde{\alpha}_t^T M_{ij}\alpha - \frac{1}{2}\widetilde{\alpha}_t^T M_{ij}\alpha_t, \quad (8)$$

where $\widetilde{\alpha} = (\alpha^\top, b)^\top$, $\widetilde{\alpha}_t$ denotes the value of $\widetilde{\alpha}$ at the t-th CCCP iteration and

$$M_{ij} = \begin{bmatrix} k_i k_j^T & \frac{1}{2}(k_i + k_j) \\ \frac{1}{2}(k_i + k_j)^T & 1 \end{bmatrix}.$$

In Eq. (5), the number of slack variables ξ_i equals to the sample number while the number of ζ_{ij} in Eq. (6) equals to the number of the constraints in set Θ_s. By introducing two variables to reduce the parameter number, i.e., $\xi = \sum_i \xi_i$ and $\zeta = \sum_{\theta_{ij}\in\Theta_s} \zeta_{ij}$, we can obtain the optimization problem at the t-th iteration of the CCCP procedure as:

$$\mathcal{J}_t = \min_{\alpha,b,\xi,\zeta} \frac{1}{2}\alpha^T G\alpha + \frac{\lambda_1}{n}\xi + \frac{\lambda_2}{n}\zeta, \quad (9)$$

which is subject to the following two convex constraints:

$$\begin{aligned} \xi &\geq \sum_i \Big(1 - (\alpha^T k_i + b) \cdot \text{sign}(\alpha_t^T k_i + b_t)\Big)_+, \\ \zeta &\geq \sum_{\theta_{ij}\in\Theta_s} \Big(\frac{1}{2}\widetilde{\alpha}^T(M_i + M_j)\widetilde{\alpha} - \widetilde{\alpha}_t^T M_{ij}\alpha + \frac{1}{2}\widetilde{\alpha}_t^T M_{ij}\alpha_t\Big)_+. \end{aligned}$$

In this way, we can obtain a sequence of sub-problems $\mathcal{J}_t$, $t = 0 \ldots t_{max}$, as in Eq. (9), each of which is essentially a Quadratic Programming (QP) problem [26]. However, the constraints are non-linear, which makes the optimization very complex. To solve the optimization problem more efficiently, the cutting plane method [27] can be applied to ease the optimization [28]. The basic idea of cutting plane method is to maintain a collection of linear constraints, substituting the original nonlinear ones.This constraint set is initialized as empty and expended progressively after obtaining a new solution. These constraints are called cutting planes. We add the obtained cutting planes into the constraint set and then use QP to solve the reduced problem. In practice, the optimization procedure halts until satisfying the ε-optimality condition, i.e., the difference between the objective value of the original problem and the objective value of the reduced cutting plane problem is smaller than a threshold ε. Assume that the cutting plane method takes at most k_{max} iterations to converge for optimizing $\mathcal{J}_t$. According to cutting

plane method, the optimum sequence $\mathcal{J}_{t,k}^*$, $k = 0 \dots k_{max}$ monotonically decreases until the convergence to the optimal solution of $\mathcal{J}_t$. The constrained large margin binary partition procedure can be summarized as in Algorithm 1.

Algorithm 1 The large margin binary partition procedure with label constraints.

1: **for** $t = 1$ **to** t_{max} **do**
2: Relax the CCCP problem into convex sub-problem $\mathcal{J}_t$ using Taylor expansion, and initialize $(\omega_{t,0}, b_{t,0}) = (\omega_t, b_t)$;
3: $k = 0$;
4: **while** ε-optimality condition is not satisfied **do**
5: Calculate the cutting planes at location $(\omega_{t,k}, b_{t,k})$ and add them into the constraint set;
6: Use QP to solve the reduced cutting-plane problem and obtain a new solution $(\omega_{t,k+1}, b_{t,k+1})$;
7: $k = k + 1$;
8: **end while**
9: **end for**

Based on the constrained large margin binary partition procedure, we can describe the induction process of SDT as in Algorithm 2. In fact, the induction of SDT is equivalent to implementing an ordinal discrimination for multiclass classification problem with the most straightforward separation performed at the root node, more sophisticated discrimination at the lower layers of the tree and so on until the leaf nodes of the tree. The proposed tree classifier architecture takes advantage of both the efficient induction of tree architecture and the high classification accuracy of large margin discriminative classifiers. Based on this architecture, the multiclass classification problem can be conducted in an optimized sequence, which thus boosts the performance of multiclass classification.

Algorithm 2 The Induction of SDT.

1: **Input:** N-class training data $\mathcal{T}$.
2: **Output:** SDT.
3: Partition $\mathcal{T}$ into two non-overlapping subsets $\mathcal{P}$ and $\mathcal{Q}$ using the large margin binary partition procedure in Algorithm 1;
4: Repeat step 3 on datasets $\mathcal{P}$ and $\mathcal{Q}$ respectively until all obtained subsets only contain training samples from a single class.

B. Prediction

Once a SDT has been constructed, it can be used to predict the class labels of the testing samples. To test a particular sample, starting at the root node, the corresponding binary discriminative function is evaluated. The node is then exited via the left edge if the value of the binary function is non-negative, or the right edge, if the value is negative. The final decision of the testing sample is given by the class label associated with the leaf node of SDT.

C. Discussion

The proposed SDT algorithm can be meant as a data driven approach for the task of multiclass classification. In fact, each binary partitioning in SDT has to choose the most appropriate separation of the classes with the largest possible margin, which explicitly exploits the structure information of data. Applying the binary partition sequentially may result in a discriminating order towards the optimality in multiclass classification. In addition, the tree architecture of SDT as well as the large margin binary partitioning procedure implemented at each node allow us to naturally optimize the error bound of the sequential discriminating, which guarantees the theoretical generalization ability of the proposed SDT algorithm as described in the Appendix.

IV. Algorithmic Analysis

In this section, we first analyze the time complexity of our proposed SDT algorithm, and then give its generalization error bound.

A. Time Complexity

Since the proposed SDT algorithm in Algorithm 2 is essentially a recursive calling of the large margin partition procedure in Algorithm 1. Therefore, we first analyze the time complexity of Algorithm 1, and then get the overall time complexity for Algorithm 2 accordingly.

In Algorithm 1, the outer iteration is CCCP, which has been shown to decrease the objective function monotonically and converge to a local minimum solution [25]. As for the inner cutting plane iteration, we have the following two theorems:

Theorem 1. *Each iteration in steps* 6-7 *of Algorithm 1 takes time* $O(sl)$ *for a constant constraint set* Ω*, where* s *is the average number of non-zero entries in* $\widetilde{\alpha}$ *and* l *is the number of two class training samples that will be partitioned by one node of SDT.*

Theorem 2. *The cutting plane iteration in steps* 4-8 *of Algorithm 1 terminates after at most* $\max\{\frac{2}{\varepsilon}, \frac{8\lambda R^2}{\varepsilon^2}\}$ *steps, where*

$$R^2 = \| \sum_i sign(\alpha_t^\top k_i + b_t)(k_i^\top, 1)^\top \|,$$

and ε *is the threshold for terminating the cutting plane iteration (see Section III-A).* $\|\cdot\|$ *denotes* ℓ_2*-norm of a vector.*

The proofs of the above two theorems are easy to verify by following the proofs in [28], and therefore are omitted here. In Figure 2, we show the convergence curve of the objective value, which is captured during the first binary partition on iris dataset using linear kernel. Here one iteration refers to one round of iterative optimization, which

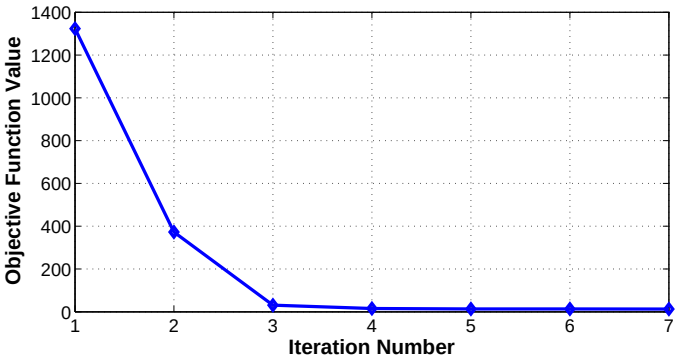


Figure 2. The convergence curve of applying Algorithm 1 in the first binary partition of iris dataset, where the linear kernel is used.

corresponds to steps 5-7 of Algorithm 1. As can be seen, the objective function converges to the minimum after about 3 iterations, which accords with the convergence analysis discussed above.

Now, we analyze the overall time complexity of Algorithm 2. As stated in Theorem 1, training a single binary partition in Algorithm 1 is observed to scale in linear time with the training set size l:

$$T_{single} = \beta l, \tag{10}$$

where β is a proportionality constant. Furthermore, assuming that the classes have the same number of training data samples, we can obtain a balanced tree in SDT using the large margin binary partition procedure in Algorithm 1. Therefore at any i-th level of the tree ($i = 0, 1, \ldots, \lfloor \log_2(N) - 1 \rfloor + 1$), the training time would be

$$T_{level_i} = 2^i \beta (\frac{n}{2^i}) = \beta n, \tag{11}$$

where N denotes the number of classes and n denotes size of the whole training set. Then the total training time for SDT becomes

$$T_{SDT} \leq \sum_{i=0}^{\lfloor \log_2(N)-1 \rfloor + 1} (\beta n) = (\lfloor \log_2(N) - 1 \rfloor + 2)\beta n. \tag{12}$$

B. Generalization Error Bound of SDT

Now we will present the generalization ability of SDT. It is worth noting that SDT forms a binary decision tree that is actually an directed acyclic graph. An important property of the directed acyclic graph based decision strategy is that an error bound can be maintained, which thus guarantees the generalization ability. The following theorem with proof omitted presents the error bound via VC analysis of directed acyclic graph [6], using the results derived in [7].

Theorem 3. *Suppose we are able to classify a random n sample of labeled examples using a directed acyclic graph on N classes containing K decision nodes with margins γ_i at node i, then we can bound the generalization error with probability greater than $1 - \delta$ to be less than*

$$\frac{130R^2}{n}\big(D^{'} log(4en) log(4n) + log\frac{2(2n)^{N-1}}{\delta}\big),$$

where $D^{'} = \sum_{i=1}^{K} \frac{1}{\gamma_i^2}$, e is the Napierian base, and R is the radius of a ball containing the support of the distribution.

As for our large margin binary partition, let ω_i and b_i be the partition parameters correctly splitting the training samples at the i-node of SDT. We define the margin of i-node in SDT to be $\gamma_i = \min_{c_i(\mathbf{x})=+1,-1}\{|\omega_i^\top \mathbf{x} + b_i|\}$, where $c_i(\mathbf{x})$ is the partition associated to training sample $\mathbf{x}$, and its value $+1$ or -1 represents which sides of the binary hyperplane the sample x should be located at. The above theorem implies that the error bound of directed acyclic graph can be constrained within an expected interval if we can enlarge the classification margin of each node classifier. Note that the individual nodes of SDT are actually a set of binary classifiers with the largest possible margins, thus the overall discriminating error of SDT is theoretically bounded, which consequentially ensures the generalization ability of SDT.

V. Experimental Results

In this section, the proposed SDT sequential multiclass classification algorithm is evaluated on a toy data (Section V-A), various benchmark datasets (Section V-B), a real world image dataset (Section V-C) and a real world text dataset (Section V-D). In each task, the performances of the following six popular multiclass classification algorithms are compared. For SVM related algorithms, we use the LIBSVM toolbox [29] for the implementation.

- OVA SVM. We train N binary SVM classifiers to separate each class from the other $N-1$ classes. As the algorithm solves several binary SVMs, for each model we assume the penalty parameter C of all binary classifiers are the same and its best parameter setting is determined via cross validation.
- OVO SVM. On each dataset, a number of $N(N-1)/2$ binary SVM classifiers are constructed to separate any pair of the N classes. We use the same experimental routine as that in OVA SVM classification, i.e., the penalty parameter C is set the same for all binary SVM classifiers on each data set.
- DAGSVM. The training phase of DAGSVM is the same as OVO SVM and thus we employ the same experimental setting.
- C&S SVM. Here "C&S" means the multiclass SVM algorithm proposed by [8]. An all-together optimization formulation is solved to tackle the multiclass classification problem, and the best penalty parameter C is determined via cross validation.
- Hierarchical SVM. At each node of the hierarchical tree, the k-means clustering algorithm [30] is employed

Table I
STATISTIC OF THE DATASETS.

Dataset	#training/testing data	#class	#dim.
iris	150/0	3	4
glass	214/0	6	13
vowel	528/0	11	10
vehicle	846/0	4	18
segment	2310/0	7	19
satimage	4435/2000	6	36

to divide the training samples into 2 non-overlapping subsets. Then a binary SVM classifier is trained as the node classifier. The best penalty parameter C for all binary SVMs is set the same via cross validation.

- SDT. The parameters λ_1 and λ_2 are empirically selected on each dataset through cross validation.

A. Toy Problem

In this subsection, we use a toy problem to illustrate that the proposed SDT algorithm is able to achieve the optimal discriminating order for multiclass classification. As shown in Figure 3(a), the toy problem is a 4-class classification task in which each class consists of 60 2-D data points represented with a specific shape. We use different multiclass classification algorithms with linear kernel to classify the 4-class data and observe the discriminating order for each algorithm. As the discriminating orders for the non-hierarchical algorithms including OVO SVM, OVA SVM and DAGSVM are arbitrary, here we only present the result for OVA SVM due to space limit. The results are shown in Figure 3(b) to Figure 3(d). From the subfigures, we can have the following observations. 1) The discriminating order of hierarchical SVM heavily relies on the clustering result of the current classes, resulting in less effective discriminating power. 2) The discriminating order is totally ignored in the OVA SVM algorithm, which affects its discriminating ability to the given toy problem. 3) The proposed SDT algorithm is able to classify the data in an optimal order in which the most separable class clusters are discriminated by the first large margin binary classifier, followed by the further separation of the remaining classes with two more classifiers. This demonstrates the effectiveness of our proposed SDT algorithm in determining the optimal discriminating order in multiclass classification.

B. Benchmark Tasks

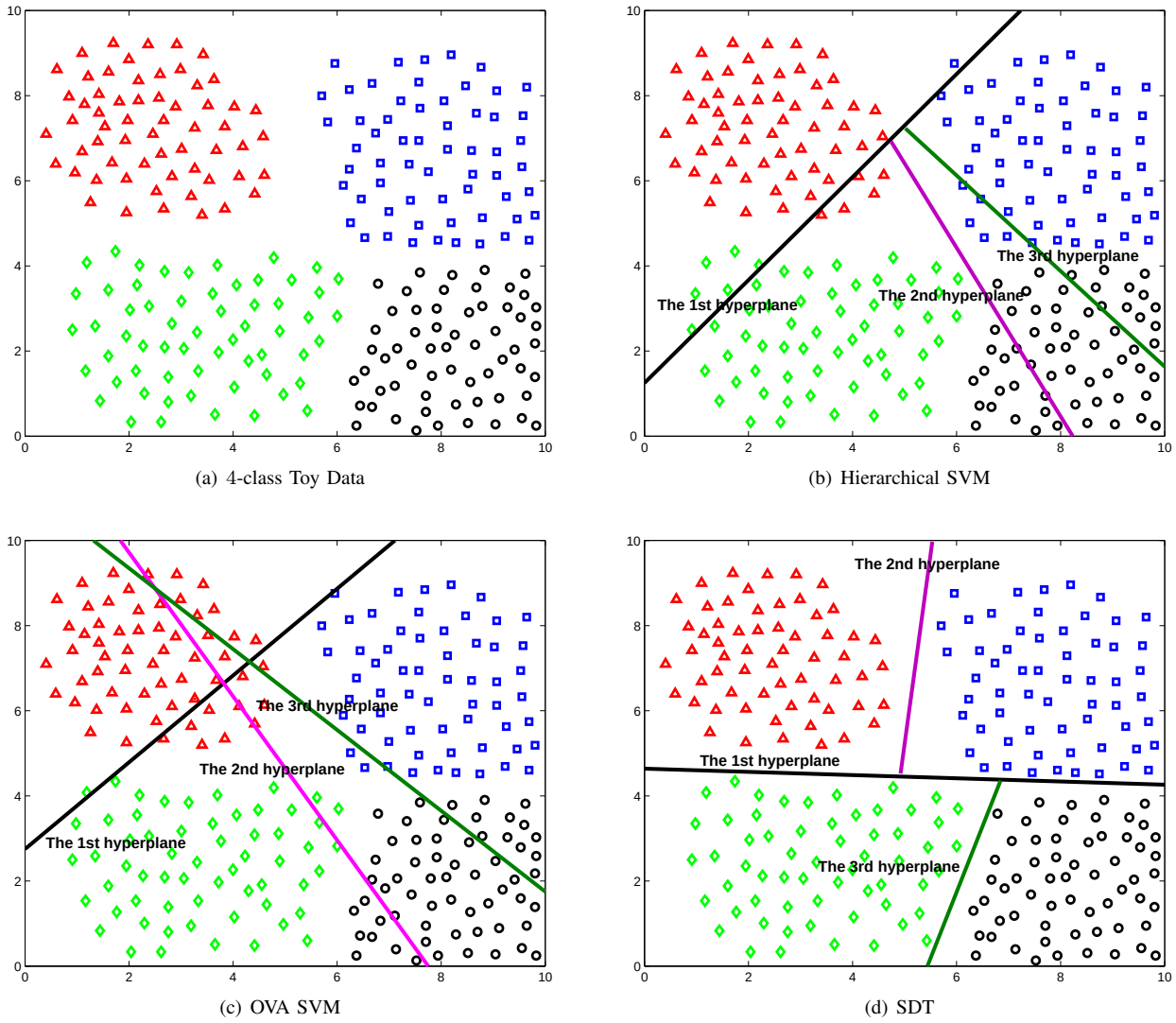
In this subsection, we evaluate the proposed SDT algorithm on six benchmark datasets popularly used in the studies of multiclass classification problem, including iris, glass, vowel, vehicle, segment and satimage [31]. The detailed statistics of these datasets are shown in Table I. Note that only for satimage, the testing data set is available, while for the other five datasets, only training data sets are available. For each dataset, we scale the training data to be in $[-1, 1]$, and then adjust the testing data to $[-1, 1]$ accordingly, if the corresponding testing data set is available [29].

We employ the classification accuracy rate as the performance evaluation metric of different algorithms. To fairly evaluate these six algorithms, we seek the best parameters on each dataset for each algorithm by performing cross validation. To train each of the six algorithms, we select two kinds of kernel functions including the parameter-free linear kernel $K(u, v) = \langle u, v\rangle$ and the RBF kernel $K(u, v) = \exp(-\gamma\|u - v\|^2)$, where the value of parameter γ is selected from $\{2^{-5}, 2^{-4}, \ldots, 2^4, 2^5\}$. For each of the SVM based algorithms, the choice of the penalty parameter C is determined from $\{2^{-5}, 2^{-4}, ..., 2^4, 2^5\}$. We use two different criteria to estimate the generalized accuracy. For satimage where both training and testing sets are available, for each parameter C (for linear kernel) or parameter pair (γ, C) (for RBF kernel), the validation performance is measured by training 70% of the training data and testing the other 30% of the training data. Then we train the whole training set using the parameter (or parameter pair) that achieves the best validation rate and predict on the testing set. For the other five datasets where the testing data are not available, we simply conduct a 10-fold cross-validation on the whole training data and report the best cross-validation rate. Note that this evaluation strategy is adopted by most of the state-of-the-art multiclass classification algorithms on the same benchmark datasets [9], [6]. To coincide with these algorithms, we also employ this evaluation strategy, and do not report the means and standard deviations obtained from multiple trainings/testings on each dataset (See Section V-C). For the proposed SDT algorithm, we select the values for parameter λ_1 and λ_2 from $\{100, 200, \ldots, 900, 1000\}$ through 10-fold cross validation, and then report the validation rate as the final result of each dataset.

In Table II, we present the classification results using the linear kernel and the RBF kernel, where each row shows the accuracy rates of different algorithms on one dataset and the best performance is indicated with bolded font. From these results, we can have the following observations. 1) The accuracy rates are similar for all five SVM based algorithms, in which the OVO SVM algorithm performs relatively better. 2) Overall the RBF kernel produces better accuracy than the linear kernel, which reveals the necessity of using nonlinear kernels in real applications. 3) The proposed SDT algorithm achieves higher classification accuracy rates compared to the other state-of-the-art multiclass classification algorithms over most of the datasets, which confirms the effectiveness of seeking the optimal discriminating order in multiclass classification.

C. Image Categorization

Image categorization is one of the natural applications of multiclass classification. To evaluate the proposed algorithm on image categorization task, we use the COREL image



(a) 4-class Toy Data (b) Hierarchical SVM

(c) OVA SVM (d) SDT

Figure 3. First three classification hyperplanes obtained with different multiclass algorithms, which shows the discriminating order adopted by different algorithms.

dataset with 25 categories, where each category has 100 images [32]. For each image, we extract a 255-dimensional color moment vector as its low level feature representation.

On the COREL image dataset, images within each category are randomly partitioned into two equal partitions. One is used for training and the other for testing. Each experiment is repeated five times based on five random splits, and the average results are reported. We use linear kernel and RBF kernel to train each of the comparison algorithms. For RBF kernel parameter γ, we select its value from $\{2^{-5}, 2^{-4}, \ldots, 2^{4}, 2^{5}\}$. For the five SVM based algorithms, the penalty parameter C is selected from $\{2^{-5}, 2^{-4}, ..., 2^{4}, 2^{5}\}$. For each parameter or parameter pair, we validate the performance of each algorithm by training on 70% of the training data and testing over the other 30% of the training data. Then the best parameter that achieves the best validation rate is adopted to train a classifier on the whole training data, which is used for further prediction on the testing set. For the proposed SDT sequential classification algorithm, the parameter λ_1 and λ_2 are selected from $\{100, 200, \ldots, 900, 1000\}$, respectively.

Table III
ACCURACIES (%) ON THE IMAGE CATEGORIZATION TASK. THE BEST PERFORMANCE IS BOLDED.

Linear kernel	*accuracy*	RBF kernel	*accuracy*
OVA SVM	66.79 ± 2.13	OVA SVM	70.12 ± 3.31
OVO SVM	71.17 ± 2.25	OVO SVM	75.81 ± 3.62
DAGSVM	69.09 ± 2.74	DAGSVM	75.55 ± 3.63
C&S	68.59 ± 2.16	C&S	73.86 ± 3.03
HierSVM	70.12 ± 2.37	HierSVM	72.27 ± 2.96
SDT	$\mathbf{73.26 \pm 1.98}$	SDT	$\mathbf{77.25 \pm 3.09}$

Table II
ACCURACY (%) ON THE BENCHMARK DATASETS USING THE LINEAR KERNEL (UPPER ROWS) AND USING THE RBF KERNEL (LOWER ROWS). THE BEST PERFORMANCE ON EACH DATASET IS BOLDED.

Linear Kernel	OVA SVM	OVO SVM	DAGSVM	C&S SVM	Hierarchical SVM	SDT
iris	96.00	97.33	96.67	96.67	97.33	**98.00**
glass	60.28	66.82	61.23	64.95	65.32	**68.49**
vowel	50.95	80.49	81.03	82.57	81.01	**83.75**
vehicle	78.72	81.09	80.13	78.72	79.82	**82.83**
segment	92.47	95.24	94.38	95.37	93.83	**97.75**
satimage	80.35	85.50	86.30	85.15	**87.15**	86.20
RBF Kernel	OVA SVM	OVO SVM	DAGSVM	C&S SVM	Hierarchical SVM	SDT
iris	96.67	97.33	96.67	96.67	97.33	**98.00**
glass	71.76	71.47	72.96	70.87	72.61	**73.16**
vowel	97.79	98.93	98.26	98.85	98.18	**97.02**
vehicle	86.63	86.64	86.32	87.12	86.13	**87.26**
segment	96.78	97.13	97.24	96.88	97.16	**97.53**
satimage	91.45	91.30	91.25	92.35	92.10	**92.45**

Table III lists the results of different algorithms on the image categorization task, where both mean and standard derivation of each algorithm are reported. From the table, we can see that the proposed SDT algorithm also outperforms the other multiclass classification algorithms, which further validates that the discriminating order plays an important role in the multiclass classification task.

D. Text Categorization

We derive a text categorization dataset from the *20 Newsgroups* corpus popularly adopted in text categorization task [33]. For each category, we randomly select 100 samples and thus form a dataset with $2,000$ documents. Note that each document is represented as a $62,061$ dimension feature vector, making the classification task extremely challenging.

We use the same experimental routine as that in image categorization, namely, the parameter γ is determined from $\{2^{-5}, 2^{-5}, \ldots, 2^4, 2^5\}$. For the SVM related algorithms, the optimal penalty parameter C is selected from $\{2^{-5}, 2^{-4}, ..., 2^4, 2^5\}$. The parameters are validated with $70\%/30\%$ split of the training data, and the parameter achieving the best performance is utilized to train a classifier on the whole training data. The parameters λ_1 and λ_2 in SDT are empirically validated from $\{100, 200, \ldots, 900, 1000\}$, respectively.

Table IV
ACCURACIES (%) ON THE TEXT CATEGORIZATION TASK. THE BEST PERFORMANCE IS BOLDED (STATISTICAL SIGNIFICANCE EXAMINED VIA PAIRWISE T-TESTS AT 95% SIGNIFICANCE LEVEL).

Linear Kernel	$accuracy$	RBF Kernel	$accuracy$
OVA SVM	51.93 ± 5.72	OVA SVM	52.83 ± 5.93
OVO SVM	57.23 ± 6.82	OVO SVM	60.05 ± 2.74
DAGSVM	59.00 ± 6.79	DAGSVM	67.67 ± 3.67
C&S	55.34 ± 6.26	C&S	66.75 ± 2.96
HierSVM	61.71 ± 5.51	HierSVM	68.26 ± 2.43
SDT	$\mathbf{63.23 \pm 5.27}$	SDT	$\mathbf{68.72 \pm 3.04}$

From Table IV, we can see that the proposed SDT algorithm clearly outperforms other multiclass algorithms in text categorization task.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have introduced a sequential discriminating tree (SDT) algorithm towards the optimal discriminating order in multiclass classification. The proposed algorithm is motivated by a theoretical bound, which guarantees its generalization ability. Experimental results demonstrate that the proposed SDT algorithm outperforms the state-of-the-art multiclass classification algorithms on a wide range of classification tasks.

The proposed algorithm has a potentially large number of applications in various areas where multiclass data are considered. Besides multiclass classification, the idea of seeking the optimal learning order can also be applied to many other learning scenarios such as unsupervised clustering, multiclass active learning and so on.

VII. ACKNOWLEDGEMENT

We would like to acknowledge to support of "NExT Research Center" funded by MDA, Singapore, under the research grant: WBS:R-252-300-001-490.

REFERENCES

[1] K. Nigam, A. McCallum, S. Thrun, and T. M. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning*, vol. 39, no. 2/3, pp. 103–134, 2000.

[2] O. Chapelle, P. Haffner, and V. Vapnik, "Support vector machines for histogram-based image classification," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1055–1064, 1999.

[3] C. H. Q. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," *Bioinformatics*, vol. 17, no. 4, pp. 349–358, 2001.

[4] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, September 1998.

[5] U. H.-G. Kreßel, "Pairwise classification and support vector machines," in *Advances in kernel methods: support vector learning*, 1999, pp. 255–268.

[6] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin dags for multiclass classification," in *Proceedings of Neural Information Processing Systems*, 1999, pp. 547–553.

[7] K. P. Bennett, N. Cristianini, J. Shawe-taylor, and D. Wu, "Enlarging the margins in perceptron decision trees," vol. 41, pp. 295–313, 1999.

[8] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," in *Proceedings of the 13th Annual Conference on Computational Learning Theory*, 2000, pp. 35–46.

[9] V. Vural and J. G. Dy, "A hierarchical method for multi-class support vector machines," in *Proceedings of International Conference on Machine Learning*, 2004, pp. 175–183.

[10] J. Weston and C. Watkins, "Support vector machines for multi-class pattern recognition," in *Proceedings of the 7th European Symposium On Artificial Neural Networks*, 1999, pp. 219–224.

[11] T.-F. Wu, C.-J. Lin, and R. C. Weng, "Probability estimates for multi-class classification by pairwise coupling," *J. Mach. Learn. Res.*, vol. 5, pp. 975–1005, December 2004.

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

[13] Y. Lee, Y. Lin, 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*, vol. 99, pp. 67–81, 2004.

[14] S. Kumar, J. Ghosh, and M. M. Crawford, "Hierarchical fusion of multiple classifiers for hyperspectral data analysis," *Pattern Analysis and Applications*, vol. 5, no. 2, pp. 210–220, 2002.

[15] E. Frank and S. Kramer, "Ensembles of nested dichotomies for multi-class problems," in *Proceedings of International Conference on Machine Learning*, 2004.

[16] L. Dong, E. Frank, and S. Kramer, "Ensembles of balanced nested dichotomies for multi-class problems," in *Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Database*, 2005, pp. 84–95.

[17] O. Pujol, P. Radeva, and J. Vitria, "Discriminant ecoc: A heuristic method for application dependent design of error correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, pp. 1007–1012, June 2006.

[18] S. Escalera, D. M. J. Tax, O. Pujol, P. Radeva, and R. P. W. Duin, "Subclass problem-dependent design for error-correcting output codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, pp. 1041–1054, June 2008. [Online]. Available: http://portal.acm.org/citation.cfm?id=1399104.1399439

[19] F. Chang, C.-Y. Guo, X.-R. Lin, and C.-J. Lu, "Tree decomposition for large-scale svm problems," *Journal of Machine Learning Research*, vol. 11, pp. 2935–2972, 2010.

[20] Y. Hu, J. Wang, N. Yu, and X.-S. Hua, "Maximum margin clustering with pairwise constraints," in *ICDM*, 2008, pp. 253–262.

[21] Y. Mu, J. Shen, and S. Yan, "Weakly-supervised hashing in kernel space," in *CVPR*, 2010, pp. 3344–3351.

[22] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[23] B. Schölkopf, R. Herbrich, and A. Smola, "A generalized representer theorem," in *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*, 2001, pp. 416–426.

[24] A. Yuille and A. Rangarajan, "The concave-convex procedure," *Neural Computation.*, vol. 15, no. 4, pp. 915–936, 2003.

[25] A. Vishwanathan, A. Smola, and S. Vishwanathan, "Kernel methods for missing variables," in *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.

[26] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[27] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[28] T. Joachims, "Training linear svms in linear time," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 217–226.

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

[30] Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classification," *Biometrics*, vol. 21, pp. 768–780, 1965.

[31] C. B. D. Newman and C. Merz, "UCI repository of machine learning databases," 1998.

[32] H. Müller, S. Marchand-Maillet, and T. Pun, "The truth about corel - evaluation in image retrieval," in *Proceedings of the International Conference on Image and Video Retrieval*, 2002, pp. 38–49.

[33] K. Lang, "Newsweeder: Learning to filter netnews," in *Proceedings of International Conference on Machine Learning*, 1995, pp. 331–339.