Hierarchical Document Categorization with Support Vector Machines

Lijuan Cai
Department of Computer Science,
Brown University
Providence, RI 02806 USA
ljcai@cs.brown.edu

Thomas Hofmann*
Department of Computer Science
Brown University
Providence, RI 02806 USA
th@cs.brown.edu

ABSTRACT

Automatically categorizing documents into pre-defined topic hierarchies or taxonomies is a crucial step in knowledge and content management. Standard machine learning techniques like Support Vector Machines and related large margin methods have been successfully applied for this task, albeit the fact that they ignore the inter-class relationships. In this paper, we propose a novel hierarchical classification method that generalizes Support Vector Machine learning and that is based on discriminant functions that are structured in a way that mirrors the class hierarchy. Our method can work with arbitrary, not necessarily singly connected taxonomies and can deal with task-specific loss functions. All parameters are learned jointly by optimizing a common objective function corresponding to a regularized upper bound on the empirical loss. We present experimental results on the WIPO-alpha patent collection to show the competitiveness of our approach.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and retrieval—*Information Filtering*; I.2.6 [**Artificial Intelligence**]: Learning—*Induction*; I.5.1 [**Pattern Recognition**]: Models—*Statistical*

General Terms

Algorithms, Experimentation, Theory

Keywords

taxonomy, document categorization, SVM, hierarchical loss, class relationship, subspace optimization

*Part of this research has been conducted while the author was visiting scientist at the Max-Planck Institute for Biological Cybernetics, Tübingen, Germany

CIKM'04, November 8–13, 2004, Washington, DC, USA.

1. INTRODUCTION

Document categorization is a crucial and well-proven instrument for organizing large volumes of textual information. Being no brainchild of our age, comprehensive classification systems have been developed and maintained by librarians since the 19th century and are today in widespread use. The advent of the Web and the enormous growth of digital content in intranets, databases, and archives, have further increased the demand for categorization. In the face of the pace and complexity of this process, manual categorization often lacks economic efficiency and automatic tools are indispensable to supplement human efforts.

In most cases, the use of statistical or machine learning techniques has been proven to be successful in this context, since it is typically more feasible to induce categorization rules based on example documents, than to elicit such rules from domain experts. The wide range of methods applied to this problem include nearest neighbor classifiers [23], neural networks [19], generative probabilistic classifiers [6, 7], and – more recently – boosting [12] and Support Vector Machines (SVMs) [5], to name just a few. Extensive experimental comparisons (e.g. [5, 24, 1]) have evidenced that among the methods available today, SVMs are highly competitive in their classification accuracy and can therefore be considered the state-of-the art in document categorization.

A potential drawback of all of the above mentioned classification methods is that they treat the category structure as 'flat' and that they do not consider relationships between categories, which are commonly expressed in concept hierarchies or taxonomies. Such structures, however, are the preferred way in which concepts, subject headings, or categories are arranged in practice. Taxonomies offer clear advantages in supporting tasks like browsing, searching or visualization. They are also easier to maintain and alleviate the manual annotation process. This is witnessed by the fact that virtually all real world classification systems have complex hierarchical structure. This includes traditional systems like Dewey or Library of Congress subject headings, the International Patent Classification (IPC) scheme [21], the Medical Subject Headings (MeSH) maintained by NIH, as well as Web catalogs created by Yahoo!, the Open Directory Project (DMOZ) or LookSmart, to name some of the most important ones.

There is reason to believe that taxonomies offer valuable information which learning methods should be able to capitalize on, in particular since the number of training examples for individual classes may be relatively small when dealing

with tens of thousands of classes. The potential loss of valuable information suffered from ignoring class hierarchies has been pointed out many times before and has led to a number of approaches that deal with ways to exploit hierarchies, e.g. [6, 8, 19, 18, 4].

In this paper, we present a generalization of the SVM classification architecture [17] that directly incorporates prior knowledge about class relationships. This is accomplished by using discriminant functions that decompose into contributions from different levels of the hierarchy. Compared to previous approaches, our main contribution is a novel formulation of hierarchical classification as a joint large margin problem (cf. Section 2), for which we derive an efficient training algorithm (cf. Section 4). Moreover, the proposed method is not restricted to the zero-one classification loss, but is able to directly incorporate specific loss functions, in particular ones derived from the taxonomy (cf. Section 3).

2. HIERARCHICAL SVM CLASSIFICATION

There are two slightly different settings for document categorization: problems involving multiple overlapping binary classes and multiclass problems. In the latter case, each document belongs to exactly one category, whereas in the former case a document may belong to multiple categories. We will present a hierarchical model for both of these settings, but will focus on the multiclass case with the experimental evaluation on the WIPO-alpha collection [22] (cf. Section 6).

2.1 Multiclass SVM Classification

We take the multiclass SVM learning approach of [3] as our starting point. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be a set of n labeled training documents. Here $\mathbf{x}_i \in \Re^d$ denotes a standard vector representation for the i-th training document. Each label y_i refers to a unique category encoded as an integer, $y_i \in \mathcal{Y} \equiv \{1, \ldots, q\}$, where q is the total number of categories.

Let us introduce a weight vector $\mathbf{w}_y$ for every class $1 \leq y \leq q$. We will refer to the stacked vector of all weights by $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_q)$. Then we can define a linear discriminant function[1]

$$F(\mathbf{x}, y; \mathbf{w}) \equiv \langle \mathbf{w}_y, \mathbf{x} \rangle \tag{1}$$

and a corresponding classification function f as

$$f(\mathbf{x}; \mathbf{w}) \equiv \operatorname*{argmax}_{y \in \mathcal{Y}} F(\mathbf{x}, y; \mathbf{w}) \,. \tag{2}$$

Eq. (2) is also known as the Winner-Take-All (WTA) rule. Generalizing from the binary classification case, the multiclass margin of a weight vector $\mathbf{w}$ with respect to an instance $(\mathbf{x}_i, y_i)$ can be defined as

$$\gamma_i(\mathbf{w}) \equiv F(\mathbf{x}_i, y_i) - \max_{y \neq y_i} F(\mathbf{x}_i, y) \tag{3}$$

Notice that the correct classification of a training instance requires a positive margin. Assuming for now that the training data can indeed be correctly classified by some weight vector $\mathbf{w}$, then we can apply the maximum margin principle to determine the weight vector $\mathbf{w}^*$ achieving optimal separation

$$\mathbf{w}^* = \operatorname*{argmax}_{\mathbf{w}: \|\mathbf{w}\|=1} \min_{i=1}^{n} \gamma_i(\mathbf{w}) \,. \tag{4}$$

This can equivalently be written as a norm minimization problem, which can also be augmented by slack variables in order to account for possible margin violations by outliers or hard instances. One arrives at the following soft-margin multiclass formulation

$$\min_{\mathbf{w}, \boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \tag{5a}$$

$$\text{s.t. } \gamma_i(\mathbf{w}) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad (\forall i) \tag{5b}$$

Notice that every non-linear constraints in Eq. (5b) can be expanded into $q-1$ linear constraints of the form

$$\langle \mathbf{w}_{y_i} - \mathbf{w}_y, \mathbf{x}_i \rangle \geq 1 - \xi_i, \quad (\forall i, y \neq y_i) \tag{6}$$

so that Eq. (5) indeed corresponds to a convex quadratic program with $n \cdot q$ linear constraints.

In the special case of $q = 2$ the constraints only involve the difference vector $\mathbf{v} \equiv \mathbf{w}_1 - \mathbf{w}_2$. Moreover it is easy to see that the penalty on $\|\mathbf{w}\|$ implies that the optimal weight vector fulfills $w_{1j} w_{2j} = 0$ for all features j. Hence, one can equivalently minimize the norm $\|\mathbf{v}\|$, showing that the $q = 2$ case indeed reduces to binary SVM classification.

2.2 SVM Learning with Class Attributes

We would like to extend the above multiclass SVM formulation to cases, where classes are not just arbitrary numbers, but can be characterized by attribute vectors $\mathbf{\Lambda}(y) \in \Re^s$. This should be carried out in a way that recovers the standard multiclass setting as a special case of an orthogonal attribute representation with $s = q$ and $\lambda_r(y) = \delta_{yr}$, i.e. a case where each class is interpreted as a binary attribute of its own. To that extent, we propose to redefine a more general version of the discriminant functions F in (1) as

$$F(\mathbf{x}, y; \mathbf{w}) \equiv \langle \mathbf{w}, \Phi(\mathbf{x}, y) \rangle \,, \tag{7}$$

where $\Phi(\mathbf{x}, y) = \mathbf{\Lambda}(y) \otimes \mathbf{x}$. Here $\otimes$ denotes a tensor product, i.e. $\Phi(\mathbf{x}, y) \in \Re^{d \cdot s}$ is a vector containing all products of coefficients from the first and second vector argument. Writing out $\Phi(\mathbf{x}, y)$ one gets

$$\Phi(\mathbf{x}, y) = \begin{pmatrix} \lambda_1(y) \cdot \mathbf{x} \\ \lambda_2(y) \cdot \mathbf{x} \\ \cdots \\ \lambda_s(y) \cdot \mathbf{x} \end{pmatrix}, \tag{8}$$

and for $\lambda_r(y) = \delta_{ry}$ this simply reduces to

$$\Phi(\mathbf{x}, y) = \begin{pmatrix} \vdots \\ 0 \\ \mathbf{x} \\ 0 \\ \vdots \end{pmatrix} \leftarrow y\text{-th position.} \tag{9}$$

Notice that in this latter case $\langle \mathbf{w}, \Phi(\mathbf{x}, y) \rangle = \langle \mathbf{w}_y, \mathbf{x} \rangle$ and Eq. (7) indeed reduces to the formulation in Eq. (1). In general, it is a straightforward consequence of the linearity of Eq. (7) to show that one can re-write F as an additive

[1] One can also introduce explicit bias terms b_y for every class, but this would complicate the presentation and leads to further complications in the optimization algorithm. We thus restrict ourselves to this simpler setting.

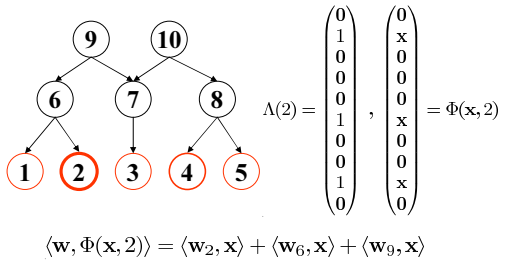


Figure 1: Taxonomy with 5 categories and a total of 10 nodes. The decomposition of the discriminant function for category 2 is depicted as an example.

superposition of linear discriminants as follows

$$F(\mathbf{x}, y; \mathbf{w}) = \sum_{r=1}^{s} \lambda_r(y) \langle \mathbf{w}_r, \mathbf{x} \rangle, \tag{10}$$

where $\mathbf{w}_r \in \Re^d$ is a weight vector associated with the r-th class attribute. Using the new definition of F in Eq. (10) to define the margin, one arrives at a quadratic program with linear constraints

$$\langle \delta\Phi_i(y), \mathbf{w} \rangle \geq 1 - \xi_i \, (\forall i, y \neq y_i) \,, \quad \xi_i \geq 0 \, (\forall i) \,, \tag{11}$$

where $\delta\Phi_i(y) \equiv \Phi(\mathbf{x}_i, y_i) - \Phi(\mathbf{x}_i, y)$.

2.3 Dual Quadratic Program

It is of conceptual and computational interest to derive the dual quadratic program (QP) of the above formulation of large margin learning with class attributes. To that extent one first forms the Lagrangian function

$$\begin{aligned} \mathcal{L}(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\zeta}) = & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \zeta_i \xi_i \\ & - \sum_{i=1}^{n} \sum_{y \neq y_i} \alpha_{iy} \left(\langle \delta\Phi_i(y), \mathbf{w} \rangle - 1 + \xi_i \right) . \end{aligned} \tag{12}$$

Computing derivatives of $\mathcal{L}$ with respect to the primal variables results in

$$\nabla_{\mathbf{w}} \mathcal{L} = 0 \iff \mathbf{w} = \sum_{i=1}^{n} \sum_{y \neq y_i} \alpha_{iy} \delta\Phi_i(y) \,, \tag{13}$$

$$\nabla_{\boldsymbol{\xi}} \mathcal{L} = 0 \iff \zeta_i = C - \sum_{y \neq y_i} \alpha_{iy}, \quad (\forall i) \,. \tag{14}$$

Since $\zeta_i \geq 0$, Eq. (14) reduces to a box constraint

$$\sum_{y \neq y_i} \alpha_{iy} \leq C, \quad (\forall i) \,. \tag{15}$$

Plugging in the optimality equation for $\mathbf{w}$ and exploiting Eq. (14) one arrives at the dual objective

$$\begin{aligned} \Theta(\boldsymbol{\alpha}) = & \sum_{i=1}^{n} \sum_{y \neq y_i} \alpha_{iy} \\ & - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{y \neq y_i} \sum_{y' \neq y_j} \alpha_{iy} \alpha_{jy'} \langle \delta\Phi_i(y), \delta\Phi_j(y') \rangle \,. \end{aligned} \tag{16}$$

The solution of the dual QP is thus characterized by

$$\boldsymbol{\alpha}^* \equiv \operatorname*{argmax}_{\boldsymbol{\alpha}} \Theta(\boldsymbol{\alpha}), \quad \text{s.t. } \alpha_{iy} \geq 0, \sum_{y \neq y_i} \alpha_{iy} \leq C \,. \tag{17}$$

Notice that the upper bound is a consequence of the introduction of shared slack variables ξ_i for every training instance. Moreover, we would like to point out that

$$\langle \Phi(\mathbf{x}_i, y), \Phi(\mathbf{x}_j, y') \rangle = \langle \boldsymbol{\Lambda}(y), \boldsymbol{\Lambda}(y') \rangle \langle \mathbf{x}_i, \mathbf{x}_j \rangle \,, \tag{18}$$

which follows immediately from the definition of Φ and thus

$$\langle \delta\Phi_i(y), \delta\Phi_j(y') \rangle = \langle \boldsymbol{\Lambda}(y_i) - \boldsymbol{\Lambda}(y), \boldsymbol{\Lambda}(y_j) - \boldsymbol{\Lambda}(y') \rangle \langle \mathbf{x}_i, \mathbf{x}_j \rangle \,. \tag{19}$$

Herein one can simply replace the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ by the values of any kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ like in standard SVM classification.

2.4 Class Attributes from Taxonomies

The application of the method presented in the previous section to classification problems with pre-defined taxonomies is straightforward. The main idea is to encode the relationship between classes, expressed in the taxonomy, in terms of a class attribute representation. We define a taxonomy as an arbitrary lattice (e.g. tree) whose minimal elements (e.g. leaves) correspond to the categories. Cases where interior nodes represent categories can be easily handled by adding a single (terminal) node to every inner node. Elements of the lattice, i.e. terminal as well as non-terminal nodes are denoted by $z \in \mathcal{Z} = \{z_1, \ldots, z_p\}$ in the following, with $p \geq q$ and where we identify $y_k = z_k$ for $k = 1, \ldots, q$.

Then we define the class attributes by

$$\lambda_z(y) = \begin{cases} v_z, & \text{if } z \prec y \\ 0, & \text{otherwise,} \end{cases} \tag{20}$$

where the relation $\prec$ denotes that a node z is a predecessor of a node y or equal to y. Here the $v_z \geq 0$ are non-negative weights, which in the simplest case can be set to 1, such that λ_z becomes an indicator function. Other choices for v_z are, for example, setting all v_z equal to a constant for nodes z at the same depth in the lattice.

Defining class attributes via common predecessors in the taxonomy leads to a very intuitive decomposition of the discriminant function into contributions from all nodes along the paths from a root to a specific leaf. Hence (7) becomes

$$F(\mathbf{x}, y; \mathbf{w}) = \sum_{z \colon z \prec y} \lambda_z(y) \langle \mathbf{w}_z, \mathbf{x} \rangle, \tag{21}$$

An illustrating example is depicted in Figure 1.

2.5 Hierarchical Multilabel Classification

The same idea can be carried out in the multilabel case of overlapping binary classes, where document may be assigned to multiple categories. While the standard learning approach amounts to treating every binary problem for class y as an independent problem with weight vector $\mathbf{w}_y$, the binary problems are coupled in the hierarchical setting. The latter happens through the shared weight vectors $\mathbf{w}_z$ for common ancestors z. Since we have not carried out experiments with this formulation, we only present a brief sketch of the formulation.

Let us assume that a set of labeled documents $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$ is given, where $\mathbf{y}_i \in \{-1, 1\}^q$ and $y_{ir} = 1$ encodes the fact that document $\mathbf{x}_i$ belongs to the r-th category. Then we can

formulate the following large margin problem:

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\sum_{r=1}^{q}\xi_{ir}, \quad \text{s.t. } \xi_{ir} \geq 0 \quad (22a)$$

$$\text{and} \quad y_{ir}\langle\Phi(\mathbf{x}_i, r), \mathbf{w}\rangle \geq 1 - \xi_{ir}, \quad \forall i, r\,. \quad (22b)$$

Notice that the (trivial) orthogonal choice of Φ in Eq. (9) 'flattens' the hierarchy and effectively leads to q independent quadratic programs, since both the objective and the constraints decompose. Other choices of Φ, in particular the form derived from a taxonomy, will not lead to a similar decomposition.

3. LOSS-SENSITIVE LEARNING

A shortcoming of the approach presented so far is that it is based on the standard misclassification loss. More precisely, it is well known that the average margin violation or hinge loss $\frac{1}{n}\sum_i \xi_i$ provides an upper bound on the empirical misclassification rate. However, in many applications the actual loss of an incorrect prediction will depend on the relation of the classes. In particular, it is reasonable to assume that confusing classes that are "nearby" in the taxonomy is less costly or severe than predicting a class that is "far away" from the correct class. Hence, we would like to work with general loss functions $\triangle : \mathcal{Y} \times \mathcal{Y} \to \Re$ where $\triangle(y, \hat{y})$ denotes the loss of predicting $\hat{y}$, when the true class is y. We assume that $\triangle(y, y) = 0$ and that $\triangle(y, \hat{y}) > 0$ for $y \neq \hat{y}$. Two problems need to be addressed: (i) how to define meaningful loss functions for taxonomies and (ii) how to modify the SVM formulation to more directly minimize (an upper bound on) the desired loss.

3.1 Hierarchical Loss Functions

The first problem of designing suitable metrics that take the position of classes in the taxonomy into account has been investigated in [18], where $\triangle(y, y')$ is defined via the length of the shortest (undirected) path connecting y and y'. Another related proposal based on similarities between categories is due to [13], but the latter does not make use of the taxonomy and rather computes the similarity between classes from the cosine-similarity between class centroid vectors.

We prefer a derivation of loss functions that is motivated from a document filtering setting, although our approach can be combined with any other loss function. As a motivation, we assume a fictive scenario in which documents are forwarded to users based on their position in the taxonomy, i.e. users may subscribe to a particular topic by specifying nodes z of interest. Denote by f_z the subscription load at a node z, i.e. the number of users that access documents based on its categorization at or below z. Moreover, denote by $c_z \geq 0$ the cost of assigning an item at or below z to a category not at or below z. Denote by $\bar{c}_z$ the cost of assigning a document to a category at or below z that actually belongs to a category not at or below z. It seems reasonable to assume that such costs can be solicited from domain experts in real-world applications. Now the loss of predicting a class $\hat{y}$ instead of y can be defined formally as

$$\triangle(y, \hat{y}) = \sum_{\substack{z: z \prec y \\ z \not\prec \hat{y}}} f_z c_z + \sum_{\substack{z: z \not\prec y \\ z \prec \hat{y}}} f_z \bar{c}_z\,. \quad (23)$$

Notice that the loss depends on the costs associated with nodes in the symmetric difference of the predecessors of the true and predicted class. In the case of a tree, the loss involves the costs for nodes on the path to the first common predecessor in the tree. For constant loads and losses this reduces to the link distance used in [18].

3.2 Cost-Sensitive Learning

It remains to generalize the presented SVM formulation to accommodate an arbitrary loss function $\triangle$. We propose to do so by scaling the penalties for margin violations proportional to the loss. This is motivated by the fact that margin violations involving an incorrect class with high loss should be penalized more severely. Implementing this idea, the cost-sensitive multiclass formulation takes the following form

$$\min_{\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \quad (24a)$$

$$\text{s.t. } \langle\mathbf{w}, \delta\Phi_i(y)\rangle \geq 1 - \frac{\xi_i}{\triangle(y_i, y)}\,, \quad (\forall i, y \neq y_i) \quad (24b)$$

$$\xi_i \geq 0, \quad (\forall i)\,. \quad (24c)$$

The linear margin constraints (24b) lead to minor modifications in the constraints of the dual problem. It is straightforward to generalize the derivation presented in Section 2, which leads to a different upper bound constraint on the dual variables. The dual problem then becomes

$$\boldsymbol{\alpha}^* \equiv \operatorname*{argmax}_{\boldsymbol{\alpha}} \Theta(\boldsymbol{\alpha}) \quad (25a)$$

$$\text{s.t. } \alpha_{iy} \geq 0\,, \quad (\forall i) \quad (25b)$$

$$\sum_{y \neq y_i} \frac{\alpha_{iy}}{\triangle(y_i, y)} \leq C\,, \quad (\forall i, y \neq y_i)\,. \quad (25c)$$

The rationale behind (24b) is that $\frac{1}{n}\sum_{i=1}^{n}\xi_i$ will now provide an upper bound on the training loss as measured by $\triangle$.

PROPOSITION 1. *Denote by $(\hat{\mathbf{w}}, \hat{\boldsymbol{\xi}})$ a feasible solution of the QP in Eq. (24). Then $\frac{1}{n}\sum_{i=1}^{n}\hat{\xi}_i$ is an upper bound on the empirical loss $\hat{\triangle} \equiv \frac{1}{n}\sum_{i=1}^{n}\triangle(y_i, f(\mathbf{x}_i))$.*

PROOF. *Notice that*

$$\hat{\xi}_i \geq \max\{0, \max_{y \neq y_i}\{\triangle(y_i, y)(1 - \langle\delta\Phi_i(y), \hat{\mathbf{w}}\rangle)\}\}\,.$$

Obviously, if $f(\mathbf{x}_i) = y_i$ then

$$\hat{\xi}_i \geq 0 = \triangle(y_i, f(\mathbf{x}_i)).$$

If $f(\mathbf{x}_i) \neq y_i$ then

$$\hat{\xi}_i \geq \triangle(y_i, f(\mathbf{x}_i))(1 - \langle\delta\Phi_i(f(\mathbf{x}_i)), \hat{\mathbf{w}}\rangle) \geq \triangle(y_i, f(\mathbf{x}_i))\,,$$

since $\langle\delta\Phi_i(f(\mathbf{x}_i)), \hat{\mathbf{w}}\rangle) \leq 0$ is implied by $f(\mathbf{x}_i) \neq y_i$. Applying the derived bound to every training example proves the claim. □

4. OPTIMIZATION ALGORITHM

4.1 General Strategy

The dual QP in Eq. (25) can become quite large in practice, since the number of α-variables equals $n \cdot (q-1)$. In particular the scaling with q, the number of categories, is problematic when compared, for instance, to standard (binary) SVMs. We propose to exploit two properties of the

dual problem in order to design a more efficient optimization algorithm.

First, notice that the upper bound constraints in the dual problem Eq. (25) factorize over the instance index. By this we mean that the constraints in Eq. (25c) do not couple dual variables α_{iy} and $\alpha_{jy'}$ belonging to different training instances i and j. This can be exploited in an optimization procedure which iteratively performs subspace optimization over all dual variables α_{iy} belonging to the same training instance. This will in general be a much smaller QP, since it freezes all α_{jy} with $j \neq i$ at their current values. This idea has also been successfully applied in the multiclass SVM optimization algorithm proposed in [3]. However, since class attribute vectors $\mathbf{\Lambda}(y)$ are in general not orthogonal, we can not use the fixpoint method proposed in [3].

Secondly, we expect the number of active constraints at the solution to be relatively small, since only a small fraction of categories $y \neq y_i$ will typically fail to achieve the required margin. As with SVMs, this is not a necessity, but for classification problems that can be solved with reasonable accuracy, this sparseness property can be observed empirically (cf. Section 6). We propose to exploit the expected sparseness by employing a variable selection strategy for the dual problem. Equivalently, this corresponds to a cutting plane algorithm for the primal QP. Intuitively, we will identify the most violated margin constraint with index (i, y) and then add the variable α_{iy} to the optimization problem. This means that we start with extremely sparse (i.e. small) problems and only successively increase the number of variables in the active set. This general approach to deal with large linear or quadratic optimization problems is also known as column selection. In practice, it is often not necessary to optimize until final convergence, which adds to the attractiveness of this approach.

4.2 Variable Selection Strategy

Since we use an iterative approach for optimization, which selects one dual variable at a time for inclusion in the sparsified optimization problem, it is important to develop a sensible strategy for selecting those variables. In particular, we would like to utilize a heuristic which focuses on those constraints that are most severely violated. In order to implement this idea, we need to quantify the extent to which constraints are violated. For that purpose, we generalize the approach of [3] for which we will present a simplified derivation in the sequel.

Given a feasible solution $\boldsymbol{\alpha}$ of the dual QP problem Eq. (25), i.e. $\boldsymbol{\alpha}$ satisfies Eq. (25b) and (25c), the necessary and sufficient conditions for $\boldsymbol{\alpha}$ to be an optimal solution are

$$\langle \mathbf{w}(\boldsymbol{\alpha}), \delta\Phi_i(y) \rangle \geq 1 - \frac{\xi_i}{\triangle(y_i, y)}\,, \quad (\forall i, y \neq y_i) \tag{26a}$$

$$\xi_i \geq 0\,, \quad (\forall i) \tag{26b}$$

$$\alpha_{iy} \left\{ 1 - \frac{\xi_i}{\triangle(y_i, y)} - \langle \mathbf{w}(\boldsymbol{\alpha}), \delta\Phi_i(y) \rangle \right\} = 0\,, \quad (\forall i, y \neq y_i) \tag{26c}$$

$$\zeta_i \xi_i = \left(C - \sum_{y \neq y_i} \frac{\alpha_{iy}}{\triangle(y_i, y)} \right) \xi_i = 0\,, \quad (\forall i)\,, \tag{26d}$$

where $\mathbf{w}(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \sum_{y \neq y_i} \alpha_{iy} \delta\Phi_i(y)$. Eq. (26c) and (26d) are KKT complimentary conditions. Let us define the following quantities for all instance and category pairs.

$$\dot{\alpha}_{iy}(\boldsymbol{\alpha}) \equiv \begin{cases} \alpha_{iy} & \text{if } y \neq y_i \\ C - \sum_{y' \neq y_i} \frac{\alpha_{iy'}}{\triangle(y_i, y')} & \text{if } y = y_i \end{cases} \tag{27}$$

$$F_{iy} \equiv \triangle(y_i, y) \left(1 - \langle \delta\Phi_i(y), \mathbf{w}(\boldsymbol{\alpha}) \rangle\right) \tag{28}$$

$$= \triangle(y_i, y) \left(1 - \sum_j \sum_{y' \neq y_j} \alpha_{jy'} \langle \delta\Phi_i(y), \delta\Phi_j(y') \rangle \right).$$

Since $\triangle(y, y) = 0$, F_{iy_i} is always 0. Notice that by using the definition of the weight vector $\mathbf{w}(\boldsymbol{\alpha})$ via the optimality condition of the primal, positive values of F_{iy} $(y \neq y_i)$ correspond to violations of the requested (functional) margin of 1. With these quantities, Eq. (26) is simplified to

$$\xi_i \geq F_{iy}\,, \quad (\forall i, y) \tag{29a}$$

$$\dot{\alpha}_{iy}(\boldsymbol{\alpha})(F_{iy} - \xi_i) = 0\,, \quad (\forall i, y) \tag{29b}$$

From Eq. (29a), we get

$$\xi_i \geq \max_y F_{iy} \tag{30}$$

By Eq. (29b), if $\dot{\alpha}_{iy}(\boldsymbol{\alpha}) > 0$, then $\xi_i = F_{iy}$. Thus

$$\xi_i = \min_{y : \dot{\alpha}_{iy}(\boldsymbol{\alpha}) > 0} F_{iy} \tag{31}$$

Notice that Eq. (31) is well defined since $\sum_{y \neq y_i} \frac{\dot{\alpha}_{iy}}{\triangle(y_i, y)} + \dot{\alpha}_{iy_i} = C$ and hence there exists at least one label y satisfying $\dot{\alpha}_{iy} > 0$ for any pattern i. Eq. (30) and (31) lead to

$$\min_{y : \dot{\alpha}_{iy}(\boldsymbol{\alpha}) > 0} F_{iy} = \xi_i \geq \max_y F_{iy}\,, \quad (\forall i). \tag{32}$$

We now define the following quantity for every instance,

$$\psi_i = \max_y F_{iy} - \min_{y : \dot{\alpha}_{iy}(\boldsymbol{\alpha}) > 0} F_{iy}\,. \tag{33}$$

It can be shown that Eq. (29) holds if and only if $\psi_i = 0$ for all i's. Therefore, we have the proposition as below.

PROPOSITION 2. *Given a feasible solution $\boldsymbol{\alpha}$ of Eq. (25), $\boldsymbol{\alpha}$ is an optimum if and only if $\psi_i = 0$ for all $i = 1, \ldots, n$.*

The proposition justifies the a selection strategy that selects training pattern for which ψ_i is maximal. Once we have selected the i-th example we select the class $y \neq y_i$ for which F_{iy} is maximal and add the variable α_{iy} to the optimization.

Moreover, the following proposition sheds some light on why it is reasonable to work on the reduced problem. Proof is skipped due to lack of space.

PROPOSITION 3. *Given a set of selected variables $S \subseteq \{(i, y) : i = 1, \ldots, n,\ y \neq y_i\}$ and a solution $\boldsymbol{\alpha}^*$ of the dual QP over the reduced problem, i.e. the problem where implicitly $\alpha_{iy} = 0$ for all $(i, y) \notin S$. Then $\boldsymbol{\alpha}^*$ is a solution to the full QP in Eq. (25) if and only if $\psi_i = 0$ for all $i = 1, \ldots, n$.*

4.3 Implementation Details

The previous discussion immediately leads to an optimization algorithm. Pseudocode is shown in Algorithm 1. The sets S_i keep track of the selected constraints for each training pattern. In step 5,6 the next constraint is selected. For the optimization in step 8 one can use any standard QP

Algorithm 1 Optimization algorithm using variable selection and subspace optimization.

inputs: training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, tolerance $\epsilon \geq 0$
initialize $S_i = \emptyset$, $\alpha_{iy} = 0$, for $i = 1, \ldots, n$, $y \neq y_i$
repeat
  compute F_{iy} from (28) and ψ_i from (33)
  select $\hat{i} = \text{argmax}_{i=1}^n \psi_i$
  select $\hat{y} = \text{argmax}_{y \neq y_{\hat{i}}} F_{\hat{i}y}$
  $S_{\hat{i}} = S_{\hat{i}} \cup \{\hat{y}\}$
  solve reduced QP over $\{\alpha_{\hat{i}y} : y \in S_{\hat{i}}\}$ [8a]
  solve reduced QP over $\bigcup_{i=1}^n \{\alpha_{iy} : y \in S_i\}$ [8b]
  $S_{\hat{i}} = S_{\hat{i}} - \{y : \alpha_{\hat{i},y} = 0\}$
until $\psi_{\hat{i}} \leq \epsilon$

solver. In order to guarantee convergence in a finite number of steps, one would need to optimize over all selected variables in step (8b). However, in practice we propose to use a variant based on step (8a) which only optimizes over the subspace of the variables in $S_{\hat{i}}$, i.e. the active dual variables belonging to the selected training instance. In order to carry out step (8a), we have used the LOQO optimization package [16] in our experiments. The tolerance ϵ specifies the termination criterion based on the maximal margin violation, although in practice one might use other heuristics to stop the training process, if one is only interested in an approximate solution.

Finally, notice that one can keep track of the quantities F_{iy} and incrementally update their values after each optimization step, since only the α_{iy} parameters for the selected i-th training instance change, while the other dual variables remain frozen at their current values. Introducing these auxiliary variables prevents the undue computational load of naively evaluating the variable selection criterion.

5. RELATED WORK

There are a number of approaches pursued in the literature that are designed to take advantage of taxonomies in document categorization. One intuitive and popular way is to use a divide-and-conquer strategy to first classify documents on a coarse level and then on successively finer level.

Koller and Sahami [6] employed this strategy in conjunction probabilistic classifier (naive Bayes [7] and slightly less naive versions thereof), which are trained at each split node in the hierarchy. The classification decision is thus decomposed into a number of local routing or refinement decisions in the taxonomy. In addition, [6] proposes feature selection at every refinement level and they show that locally only a small number of discriminative feature may be sufficient to achieve reasonable classification accuracy. Follow-up work on the feature selection aspect has been performed by Mladenik and Grobelnik [9]. The downside of this approach is the use of a less competitive classifier, naive Bayes, together with an independent training process for each refinement classifier. The latter may lead to suboptimal discrimination, since the classifiers are finally operated in a specific architecture that combines their outputs. Even more severe are the disadvantages of the greedy decision process, which does not allow to recover from incorrect routing decision made at higher levels of the hierarchy. Improved non-greedy techniques have been investigated by Charkabarti et al. [2].

Divide-and-conquer strategies in conjunction with SVM classifiers have been proposed by Dumais and Chen [4] and by Sun and Lim [13]. Again, classifiers are trained independently and their outputs are combined by integrating scores along each path. In [4] a sigmoidal transformation is applied to derive estimates of posterior probabilities. These probabilities are then either thresholded independently or combined multiplicatively and then thresholded. In [13] a heuristic is developed to select training instances for training each refinement classifier. [4] also use a feature selection strategy similar to [6].

Hierarchical neural networks architectures have been utilized by Ruiz and Srinivasan [10] as well as by Weigend, Wiener, and Pedersen [19]. In both approaches a quite aggressive feature selection and/or dimension reduction step is necessary in order to reduce the number of input weights in the neural network. The training of networks at different levels is again performed independently using back-propagation, leading to the same problems that were mentioned above.

In the context of naive Bayes classification, McCallum et al. [8] have proposed the use of shrinkage, a particular form of smoothing, to derive improved estimates of parameters for the class conditional distributions. The hierarchy is thus used to overcome sparseness problems in parameter estimation and not in a divide-and-conquer manner. Toutanova et al. [14] have developed an improved Expectation Maximization algorithm that refines the technique of [8]. The main downside of this line of work is that naive Bayes classifiers are often not competitive and the gain from using the hierarchy is often less than the loss in accuracy suffered relative to more competitive methods like SVMs.

Finally, we would like to point out that our method is a natural generalization of the multiclass SVM formulation proposed in [3, 20] and a particularly interesting special case of a more general learning architecture presented in [15].

Comparing our approach with previous work, we would like to point out what we believe to be the main benefits and strengths of our method. (i) Our architecture avoids a greedy decision process, since it is not based on successive refinements. (ii) The parameters of the model are fitted by optimizing a joint objective. (iii) Being a generalization of SVM learning, we can reasonably expect a high classification accuracy. (iv) Our methods are not restricted to trees, but can handle arbitrary lattices, which in particular includes multiple trees. (v) A sparse approximation via variable selection can be used to improve the scalability of the method. The only downside we see is the fact that we have not utilized feature selection so far, but this is a topic that can be easily addressed in future work.

6. EXPERIMENTS

6.1 Experimental Setup

In this section, we compare the standard flat multiclass SVM with our hierarchical multiclass SVM on two datasets: a synthetic data collection generated according to a hierarchical scheme and the WIPO-alpha collection [22]. The taxonomy in either collection is a tree with categories in the same depth of the tree. The particular loss function that is

used in the experiments is

$$\triangle(y,\hat{y}) = \left(\sum_{\substack{z:z\prec y\\ z\not\prec\hat{y}}}\frac{1}{2}\right) + \left(\sum_{\substack{z:z\not\prec y\\ z\prec\hat{y}}}\frac{1}{2}\right), \tag{34}$$

which is a special case of Eq. (23) with constant frequency f_z and costs c_z, $\bar{c}_z$. It also equals half the distance from y to $\hat{y}$ in case of a tree structure. The class attributes have been chosen according to Eq. (20). More specifically, we have set

$$v_z = \sqrt{\frac{1}{depth}} = const\,, \tag{35}$$

where *depth* is the depth of the tree structure. This scaling is used for mathematical convenience, since it implies that $\langle \mathbf{\Lambda}(y), \mathbf{\Lambda}(y)\rangle = 1$ and therefore we get that

$$\langle \Phi(\mathbf{x},y), \Phi(\mathbf{x},y)\rangle = \langle \mathbf{\Lambda}(y), \mathbf{\Lambda}(y)\rangle\langle \mathbf{x},\mathbf{x}\rangle = \langle \mathbf{x},\mathbf{x}\rangle\,. \tag{36}$$

This is also to facilitate fair comparison between flat and hierarchical SVM classification since in the former case $\Phi(\mathbf{x},y)$ can be viewed as in Eq. (9) and hence $\langle \Phi(\mathbf{x},y), \Phi(\mathbf{x},y)\rangle$ equals $\langle \mathbf{x},\mathbf{x}\rangle$ as well.

In all experiments a linear kernel is used, since it has computational advantages during optimization, while achieving competitive performance compared to other more complicated kernels such as polynomial kernel and RBF kernel in the context of text categorization [5]. Moreover, document vectors are normalized to be of unit length, $\|\mathbf{x}_i\|_2 = 1$, as a preprocessing step. The test set performance was computed using cross-validation and macro-averaging. The tolerance parameter ϵ in algorithm 1 is set to 0.01 by default and 0.1 for some large runs. [3] points out that the choice of $\epsilon = 0.1$ achieves a good tradeoff between running time and generalization performance.

6.2 Evaluation Measures

To evaluate the effectiveness of multi-class categorization systems, we employ four different measures: *accuracy*, *precision*, *taxonomy-based loss*, and *parent accuracy*. The former two are standard metrics commonly applied in the context of multiclass problems (e.g. [11]). However, in the case of taxonomies, it is often desirable to measure the quality of the prediction by taking the relationship between categories into account. We therefore propose the latter two metrics.

Each metric with respect to a set of test examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ is defined as follows.

1. Accuracy

   Accuracy evaluates how often the prediction $f(\mathbf{x})$ is correct. Let [.] be 1 if the predicate inside is true and 0 otherwise.

   $$\mathrm{acc}(f) = \frac{1}{n}\sum_{i=1}^{n}[f(\mathbf{x}_i) = y_i] \tag{37}$$

2. Precision

   Accuracy evaluates the quality of the top-ranked category. In reality the prediction results are sometimes viewed as a ranked list. It is thus preferred that the correct label is ranked as high as possible. One metric to evaluate the quality of the ranking is precision.

   $$\mathrm{prec}(f) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{|\{y : F(\mathbf{x}_i,y) \geq F(\mathbf{x}_i,y_i)\}|}\right). \tag{38}$$

   Put roughly, $1/\mathrm{prec}(f)$ can be regarded as the average position the true category is ranked at.

3. Taxonomy-based Loss

   As noted earlier, it is reasonable to assume different mistakes will incur different loss in a taxonomy. The loss is characterized by a loss function $\triangle$. We therefore have the taxonomy-based loss as

   $$\triangle\text{-loss}(f) = \frac{1}{n}\sum_{i=1}^{n}\triangle(y_i, f(\mathbf{x}_i))\,. \tag{39}$$

   The particular loss function we have utilized is defined in Eq. (34).

4. Parent Accuracy

   Parent accuracy measures the accuracy at the level of category's parents. Let $\tilde{\mathcal{Z}} = \{z : z \in \mathcal{Z} \wedge \exists y \in \mathcal{Y} \text{ s.t. } \mathrm{parent}(y) = z\}$. $\mathrm{parent}(y) = z$ means that z is y's immediate parent. Let $\tilde{f}(\mathbf{x}) = \mathrm{parent}(f(\mathbf{x}))$. Then the parent accuracy is defined as

   $$\mathrm{pacc}(f) = \frac{1}{n}\sum_{i=1}^{n}[\tilde{f}(\mathbf{x}_i) = \mathrm{parent}(y_i)]\,. \tag{40}$$

   Assume two algorithms have similar accuracy. If a misclassification occurs, the algorithm with higher parent accuracy then is more likely to assign an instance to the true category's siblings, than to assign it to a class that is farther away from the correct class. An algorithm with a higher parent accuracy is thus favored, for instance, when used as an automatic categorization tool to assist human experts.

6.3 Synthetic Data

The first step of generating synthetic data is to decide on the tree structure and the number of features. Then random weight vectors are generated for each node in the tree, nodes at the same depth conforming to the same multivariate normal distribution. The co-variance matrices are all chosen to be diagonal for simplicity and the variances decrease from top to bottom of the tree. The sum of the weight vectors from a category's path to the root is then assigned as the category's weight vector. Data points are randomly generated and then assigned to the category with the highest score. If the difference between the highest score and the second highest score is more than a given threshold ρ, the data point is accepted. Otherwise it is rejected.

The way the artificial data is generated assures that weight vectors of nearby categories are closer than that of distant categories, simulating a property that we expect to be relevant for real-world taxonomies. Table 1 summarizes the performance of 5-fold cross-validation with hard margin separation. We observe that the hierarchical SVM consistently excels on all runs with respect to all considered measures.

6.4 WIPO-alpha Collection

The World Intellectual Property Organization (WIPO) published the WIPO-alpha collection in 2002 [22]. The patent documents in the collection are classified according to a standard taxonomy known as International Patent Classification (IPC) [21]. IPC categories are organized in a four-level hierarchy, i.e. sections, classes, subclasses and groups

#children	depth	ρ	acc (%)		prec (%)		$\triangle$-loss		pacc (%)	
			flat	hsvm	flat	hsvm	flat	hsvm	flat	hsvm
3	3	0.001	68.9	72.7	81.7	84.2	0.621	0.505	80.1	84.4
		0.1	83.4	89.9	90.8	94.7	0.351	0.205	88.0	92.9
3	2	0.001	87.1	90.0	93.1	94.6	0.193	0.158	93.6	94.2
		0.1	97.4	98.7	98.7	99.3	0.0478	0.0236	97.9	99.0
6	2	0.001	67.5	69.3	80.2	82.0	0.513	0.465	81.1	84.2
		0.1	85.2	90.5	90.9	94.4	0.244	0.15	90.4	94.7

Table 1: Experimental comparison of flat SVM (flat) and hierarchical SVM (hsvm) on synthetic data. The number of examples is fixed to 1000. The number of features is fixed to 20. '#children' refers to the number of child nodes that each interior node has, 'depth' the depth of tree, ρ the cutoff threshold, 'acc' the accuracy, 'prec' the precision, '$\triangle$-loss' the taxonomy-based loss, and 'pacc' the parent accuracy.

section	#cat	#doc	acc (%)		prec (%)		$\triangle$-loss		pacc (%)	
			flat	hsvm	flat	hsvm	flat	hsvm	flat	hsvm
A	694	10962	42.3	**42.9**	51.7	**53.2**	1.24	**1.15**	61.5	**65.0**
B	1172	14690	33.2	**33.8**	41.5	**43.1**	1.54	**1.41**	57.3	**62.2**
C	852	16245	**35.5**	35.1	**44.8**	44.6	1.32	**1.23**	61.5	**65.6**
D	160	1710	41.8	**42.8**	52.3	**54.4**	1.20	**1.08**	65.4	**69.1**
E	230	3027	**34.7**	34.3	44.8	**46.3**	1.38	**1.30**	62.7	**64.2**
F	675	6685	31.2	**32.4**	40.6	**42.9**	1.47	**1.33**	57.6	**63.3**
G	470	10302	41.0	**41.2**	50.3	**51.1**	1.32	**1.26**	60.6	**63.0**
H	403	11629	43.0	**43.1**	54.2	**55.2**	1.12	**1.07**	63.3	**66.2**

Table 2: Performance comparison of flat SVM (flat) and hierarchical SVM (hsvm) on WIPO-alpha subtrees rooted at various section codes. '#cat' refers to the number of categories, '#doc' the total number of documents in the section, 'acc' the accuracy, 'prec' the precision, '$\triangle$-loss' the taxonomy-based loss, and 'pacc' the parent accuracy. Bold face is used to mark better performance. $\epsilon = 0.1$ for runs on section A, B, C, F, G, and H.

data	#cat	#doc	acc (%)		prec (%)		$\triangle$-loss		pacc (%)	
			flat	hsvm	flat	hsvm	flat	hsvm	flat	hsvm
A, sample 3	694	1781	10.6	**11.7**	17.3	**20.5**	2.12	**1.87**	34.9	**43.2**
B, sample 3	1172	3033	9.56	**11.3**	14.7	**18.9**	2.25	**1.99**	36.5	**45.6**
C, sample 3	852	2212	12.1	**13.3**	18.1	**20.7**	1.90	**1.69**	45.4	**53.0**
D, sample 3	160	391	19.7	**20.5**	27.2	**30.9**	1.71	**1.54**	48.9	**57.3**
E, sample 3	230	600	10.2	**11.4**	17.3	**20.6**	2.01	**1.82**	40.5	**48.3**
F, sample 3	675	1729	13.1	**14.5**	19.4	**22.8**	2.02	**1.75**	40.8	**50.5**
G, sample 3	470	1228	12.4	**13.6**	18.9	**22.4**	2.09	**1.87**	35.2	**43.5**
H, sample 3	403	1084	14.8	**15.7**	22.6	**25.0**	1.81	**1.66**	42.0	**48.0**

Table 3: Performance comparison of flat SVM and hierarchical SVM on WIPO-alpha corpus with subsampling. The notations are as those in Table 2. $\epsilon = 0.1$ for 'A, sample 3', 'B, sample 3', and 'C, sample 3'.

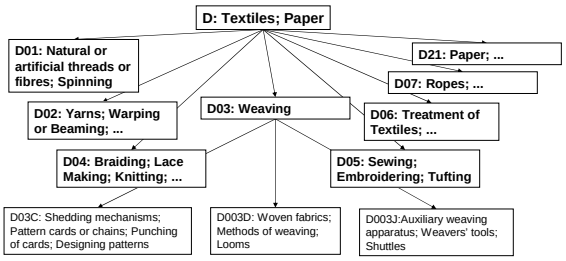


Figure 2: Part of the IPC classification hierarchy rooted at section 'D' which contains a total of 160 main groups. Only classes and subclasses for D03 are shown.

(main groups and subgroups). Part of section D is illustrated in Figure 2 for concreteness.

While a patent can have multiple categories, exactly one of them is labeled as the primary category. Predicting the primary categories therefore is a multiclass categorization problem. For our experiments we have indexed the title and claim contents. Document parsing, tokenization, and term normalization have been performed with the MindServer retrieval engine[2].

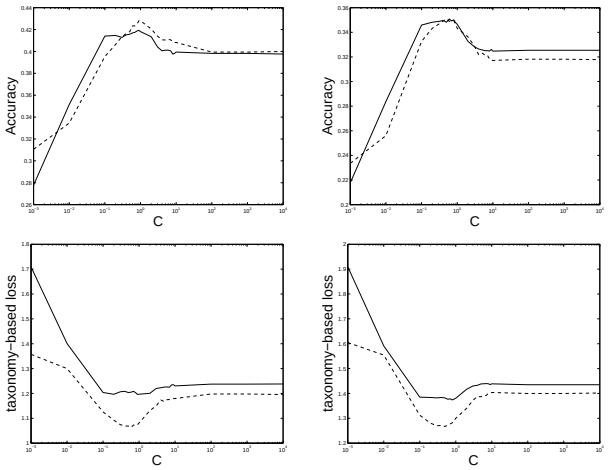
Table 2 compares the performance of the flat SVM and hierarchical SVM with respect to all 8 sections. When dealing with a specific section, only documents with their main category in the section in question are taken into account. Three-fold cross-validation has been performed for all runs. We observe that hierarchical SVM outperforms flat SVM in terms of $\triangle$-loss in all cases. This can be attributed to the fact that it explicitly optimizes an upper bound on the $\triangle$-loss of training set as well as to the specific hierarchical form of the discriminant function. Moreover, hierarchical SVM in most cases also produces higher accuracy, precision and parent accuracy.

To investigate the effect of the training size, we have furthermore subsampled the data. Results are shown in Table 3, where three samples (or all available documents, if the category possesses less than three training documents) are randomly picked for each category. Again three-fold cross-validation has been performed on all runs. We observe that hierarchical SVM outperforms flat SVM. Moreover, the performance gains are more significant in the scenario with fewer training documents. This demonstrates that the hierarchical formulation, which couples categories through the weight vectors of common ancestors, is particularly useful when operated on small training sets, since it allows more reliable estimates for weight vectors associated with higher-level nodes by effectively pooling observation in a manner similar to [8].

Following the heuristic utilized in SVMlight [5], C is set to 1 in the above runs (remember that input vectors are normalized to unit length). Our experiments show that is a good choice. Figure 3 depicts the accuracy of the investigated algorithms for varying values of C. It appears that $C = 1$ leads to a decent performance and that the hierarchical SVM always achieves a lower $\triangle$-loss during testing.

Figure 4 is an example of how the optimization process evolves over time as more and more variables are selected.

[2]http://www.recommind.com



on section D on section E

Figure 3: Performance on two sections of WIPO-alpha with varying regularization parameter C. Solid lines correspond to flat SVM and dashed lines correspond to hierarchical SVM.

For that purpose we define the active dual variable ratio as

$$\frac{|\{\alpha_{iy}|y \neq y_i \wedge \alpha_{iy} > 0\}|}{n \times (q-1)} . \quad (41)$$

We observe that in the beginning iterations, the variable selection strategy almost always add a new dual variable in each iteration. When the active set reaches a reasonable size, the growth of the active set sizes slows down and more efforts are focused on optimizing the variables that are already in the set. In all our experiments on WIPO-alpha, the learned solutions were very sparse, usually with an active ratio in the range of $[0.005, 0.1]$.

Our method of adding one or zero dual variable each time into optimization also leads to sparser solutions and faster convergence, when compared to strategies such as the one in [3] that consider all variables belonging to the same instance in the subspace optimization. Since the active set increases slowly in our case and since we restrict optimization to the variables in the active set, our method needs more subspace optimizations, but needs to solve significantly smaller QPs in every iteration. To show how the computational complexity works out in a realistic example, we have trained the hierarchical SVM on the D section of WIPO-alpha with both optimization strategies. Our approach takes about 2,200 seconds with a final fraction of 4.6% non-zero dual variables. Without maintaining an active set, the learning has taken about 42,200 seconds with a larger active ratio of 4.9%. The sparser solution can be explained by the conservative manner of constructing the active set in our approach.

7. CONCLUSIONS

We have proposed a large margin architecture for hierarchical categorization that extends the strengths of Support Vector Machine classification to also take advantage of information about class relationships encoded in a taxonomy. It is possible to minimize upper bounds on arbitrary loss functions, in particular ones that quantifies the severeness of incorrect categorizations based on the taxonomy. We be-

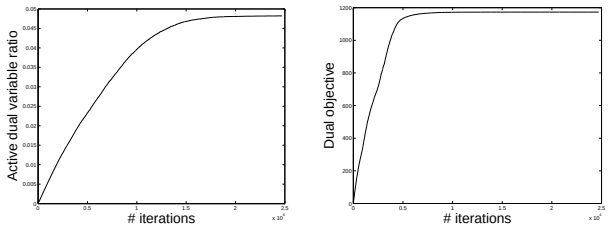


Figure 4: Optimization process of the hierarchical SVM on D section. The objective of the dual problem is defined in Eq. (16)**.**

lieve this to be a valuable feature in many real-world applications. Furthermore, we have derived a column generation algorithm to more efficiently deal with the large number of margin constraints. Results on patent classification have confirmed the competitiveness of our overall approach.

8. REFERENCES

[1] A. Cardoso-Cachopo and A. L. Oliveira. An empirical comparison of text categorization methods. In *Proceedings of the 10th International Symposium on String Processing and Information Retrieval (SPIRE'03)*, number 2857 in Lecture Notes in Computer Science, pages 183–196. Springer Verlag, 2003.

[2] S. Charkabarti, B. Dom, R. Agrawal, and P. Raghavan. Unsing taxonomy, discriminants, and signatures for navigating in text databases. In *Proceedings of the 23rd Conference on Very Large Databases (VLDB'97)*, pages 560–573, 1997.

[3] K. Crammer and Y. Singer. On the algorithmic implementation of multi-class kernelbased vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

[4] S. T. Dumais and H. Chen. Hierarchical classification of Web content. In *Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval (SIGIR'00)*, pages 256–263, 2000.

[5] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In C. Nédellec and C. Rouveirol, editors, *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, number 1398, pages 137–142. Springer Verlag, 1998.

[6] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, 1997.

[7] D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, pages 4–15, 1998.

[8] A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. Improving text clasification by shrinkage in a hierarchy of classes. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 359–367, 1998.

[9] D. Mladenić and M. Grobelnik. Feature selection for classification based on text hierarchy. In *Proceedings of the Conference on Automated Learning and Discovery*, 1998.

[10] M. E. Ruiz and P. Srinivasan. Hierarchical text categorization using neural networks. *Information Retrieval*, 5(1):87–118, 2002.

[11] R. E. Schapire and Y. Singer. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[12] R. E. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio applied to text filtering. In *Proceedings of 21th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 215–223, 1998.

[13] A. Sun and E.-P. Lim. Hierarchical text classification and evaluation. In *International Conference on Data Mining (ICDM)*, pages 521–528, 2001.

[14] K. Toutanova, F. Chen, K. Popat, and T. Hofmann. Text classification in a hierarchical mixture model for small training sets. In *Proceedings of the Tenth International ACM Conference on Information and Knowledge Management (CIKM)*, 2001.

[15] I. Tsochantardis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*, 2004.

[16] R. J. Vanderbei. LOQO: An interior point code for quadratic programming. *Optimization Methods and Software*, 11:451–484, 1999.

[17] V. Vapnik. *Statistical Learning Theory*. Wiley and Sons Inc., New York, 1998.

[18] K. Wang, S. Zhou, and S. C. Liew. Building hierarchical classifiers using class proximity. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *Proceedings of VLDB-99, 25th International Conference on Very Large Data Bases*, pages 363–374. Morgan Kaufmann Publishers, San Francisco, US, 1999.

[19] A. S. Weigend, E. D. Wiener, and J. O. Pedersen. Exploiting hierarchy in text categorization. *Information Retrieval*, 1(3):193–216, 1999.

[20] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, 1998.

[21] World Intellectual Property Organization. International patent classification. URL, 2001. http://www.wipo.int/classifications/en/.

[22] World Intellectual Property Organization. Wipo-alpha sataset. URL, 2003. http://www.wipo.int/ibis/datasets.

[23] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 13–22, 1994.

[24] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.