Hierarchical Classification via Orthogonal Transfer

Dengyong Zhou, Lin Xiao, and Mingrui Wu
Microsoft Research
Redmond, WA 98052
{denzho, lin.xiao, mingruiw}@microsoft.com

Abstract

We consider multiclass classification problems in which the set of labels are organized hierarchically as a category tree, and the examples are classified recursively from the root to the leaves. We propose a hierarchical support-vector-machine that encourages the classifiers at each node of the tree to be different from the classifiers at its ancestors. More specifically, we introduce regularizations that force the normal vector of the classifying hyperplane at each node of the tree to be orthogonal to those at its ancestors as much as possible. We establish sufficient conditions under which such an objective is a convex function of the normal vectors. We also present an efficient dual-averaging method for solving the resulting nonsmooth convex optimization problem. We evaluate the method on a number of real-world text categorization tasks and obtain state-of-the-art performance.

1 Introduction

In many multi-class classification problems, such as document and web categorization, the set of possible labels are often organized in a hierarchical structure, i.e., a category tree or a more general taxonomy. While we can always approach such problems using a generic multiclass classifier such as the multiclass Support Vector Machine (SVM) [15, 2], a really interesting question is if we can improve the classification accuracy by using the additional hierarchical structure over the categories.

One straightforward way for exploiting the hierarchical structure is to decouple the problem into a set of independent classification problems, each defined for an internal node in the hierarchy for classification between its immediate subclasses [6, 13, 4]. To better exploit the semantic relationship embedded in the hierarchy, some researchers imposed statistical similarity constraints between the probabilistic models for adjacent nodes in the hierarchy (e.g., [8]). Similarly, several work on multi-task and transfer learning employed hierarchy-induced regularizations that try to make the classifiers at adjacent nodes as close as possible [1, 5].

Another popular methodology for hierarchical classification is to use tree-induced loss functions (e.g., [1, 3]). Roughly speaking, a tree-induced loss for misclassifying two classes is proportional to the length of the undirected path connecting these two classes (i.e., their graph distance). A closely related approach is to embed the category tree into a Euclidean space such that two classes connected by a shorter path stay closer in the embedding space [14]. Using tree-induced losses captures the idea that misclassifications between similar classes with shorter graph distances are less severe thus should receive less penalty. On the other hand, it is also generally true that classification between such similar classes are much harder than classifying classes with longer graph distances. In a sense, this approach does not deal with the hardness of classifying similar classes at lower levels in the hierarchy, but rather downplay the difficulty by assigning them small penalties.

In this paper, we develop a hierarchical classificaton method that directly tackle the difficulty of classifying very similar classes at lower levels of the hierarchy. In particular, our formulation encourages the classifier at each node be different from the classifiers at its ancestors as much as possible. The key observation is that the semantic relationship among the categories in a hierarchical structure are usually of the type of generalization-specialization. In other words, the lower level categories are supposed to have the same general properties as the higher level categories plus additional more specific properties (a similar observation was made in [6]). For example, for classification between documents on `sports` and `computer science`, the frequency of the word `computer` is a very indicative feature. However, between the two subclasses `compiler` and `operating system` in `computer science`, the word `parsing` can be much more indicative than `computer`. In general, classifications at different levels of the hierarchy may rely on very different features, or different combinations of the same features.

In the context of hierarchical SVM, we formalize the above observation by using regularizations that force the normal vector of the classifying hyperplane at each node to be orthogonal to those at its ancestors as much as possible. We show that under sensible conditions on the regularization weights, training such a hierarchical SVM is a convex optimization problem. We also develop a variant of the dual-averaging method [9, 16] which is very efficient for solving such problems. In preliminary experiments on a number of real-world text categorization tasks, our method out-performs current state-of-the-arts for hierarchical classification.

2 Problem Setting

Let $\mathcal{X} \subset \mathbf{R}^n$ be the instance domain and let $\mathcal{Y}$ be the set of labels. In the setting of hierarchical classification, the labels in $\mathcal{Y}$ are identified as nodes in a category tree. Let $L = |\mathcal{Y}|$. Without loss of generality, we assume $\mathcal{Y} = \{0, 1, \ldots, L-1\}$ and let 0 be the root of the tree. For each node $v \in \mathcal{Y}$, denote by $\mathcal{C}(v)$ the set of children of v, $\mathcal{S}(v)$ the set of siblings of v, $\mathcal{A}(v)$ the set of ancestors of v (excluding itself), and $\mathcal{D}(v)$ the set of descendants of v (excluding itself). For convenience, we also define $\mathcal{A}^+(v) = \mathcal{A}(v) \cup \{v\}$ and $\mathcal{D}^+(v) = \mathcal{D}(v) \cup \{v\}$.

Let $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$ be a set of training examples, where each $\mathbf{x}_i \in \mathcal{X}$ and each $y_i \in \mathcal{Y}$. Our goal is to learn a classification function $f : \mathcal{X} \to \mathcal{Y}$ that attains a small classification error. We associate each node $v \in \mathcal{Y}$ with a vector $\mathbf{w}_v \in \mathbf{R}^n$, and focus on classifiers $f(\mathbf{x})$ determined by the following recursive procedure:

$$f(\mathbf{x}) = \left\{ \begin{array}{l} \textbf{initialize } v := 0 \\ \textbf{while } \mathcal{C}(v) \text{ is not empty} \\ \qquad v := \underset{u \in \mathcal{C}(v)}{\operatorname{argmax}} \ \mathbf{w}_u^\top \mathbf{x} \\ \textbf{return } v \end{array} \right\}. \tag{1}$$

In other words, an instance is labeled by sequentially choosing the category of which the associated vector outputs the largest score among its siblings till a leaf node is reached. This classifier always return a leaf node. For a testing example $(\mathbf{x}, y)$ where the label y is not a leaf node, a classification error is declared if and only if $y \notin \mathcal{A}^+(f(\mathbf{x}))$.

3 Hierarchical SVM with Orthogonal Transfer

In this section, we describe a hierarchical SVM for training classifiers of the form (1). It is clear that the task of learning $f(\mathbf{x})$ is reduced to learning the set of vectors $\{\mathbf{w}_v \,|\, v \in \mathcal{Y}\}$, which correspond to the normal vectors of the classifying hyperplanes.

As explained in the introduction, our method is motivated by the observation that accurate classifications at different levels of the hierarchy may rely on very different features, or different combinations of the same features. In order to capture such effects, we use the regularization terms $|\mathbf{w}_u^\top \mathbf{w}_v|$ whenever $u \in \mathcal{A}(v)$. These regularizations force each normal vector to be orthogonal to those at its ancestors as much as possible. More specifically, we propose to solve the optimization problem

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} \sum_{v \in \mathcal{Y}} k(v,v) \|\mathbf{w}_v\|^2 + \sum_{v \in \mathcal{Y}} \sum_{u \in \mathcal{A}(v)} k(u,v) |\mathbf{w}_u^\top \mathbf{w}_v| + \frac{C}{m} \sum_{i=1}^m \xi_i \\ \text{subject to} & \mathbf{w}_v^\top \mathbf{x}_i - \mathbf{w}_u^\top \mathbf{x}_i \geq 1 - \xi_i, \ \forall\, u \in \mathcal{S}(v), \ \forall\, v \in \mathcal{A}^+(y_i), \ \forall\, i \in \{1, \ldots, m\}, \\ & \xi_i \geq 0, \ \forall\, i \in \{1, \ldots, m\}. \end{array} \tag{2}$$

Here the optimization variables are the normal vectors $\mathbf{w}_v$ for all $v \in \mathcal{Y}$ and the slack variables ξ_i for all $i \in \{1, \ldots, m\}$. Both the pairwise function $k : \mathcal{Y} \times \mathcal{Y} \to \mathbf{R}_+$ (a nonnegative matrix) and the constant C are parameters that need to be selected before solving the above optimization problem.

We have the following remarks on the formulation (2):

- In the constraints, each example $(\mathbf{x}_i, y_i)$ is used for discriminating its category y_i and all its ancestors from their own siblings. Classifier pairs that do not have a common parent do not appear together in a constraint. This reflects the recursive nature of the classifier (1).
- We use the same slack variable ξ_i for all discriminative constraints associated with the example $(\mathbf{x}_i, y_i)$. Nevertheless, the classification margins at different levels in the hierarchy can be effectively differentiated by setting the diagonal coefficients $k(v,v)$.
- Since $\mathbf{w}_0$ does not appear in the constraints (because the root does not have any sibling), it is clear that its optimal value is the all-zero vector, and we never need it for classification.

3.1 Convexity

We give a sufficient condition on the nonnegative pairwise function k such that the formulation (2) is a convex optimization problem. For convenience, we assume k is *symmetric*, i.e., $k(u,v) = k(v,u)$ for all $u, v \in \mathcal{Y}$. Moreover, for two different nodes u and v, let $k(u,v) = 0$ whenever u is neither an ancestor nor a descendant of v. It suffices to establish the convexity of the function

$$\Omega(\mathbf{w}) = \frac{1}{2}\left(\sum_{v\in\mathcal{Y}} k(v,v)\|\mathbf{w}_v\|^2 + \sum_{u\neq v} k(u,v)|\mathbf{w}_u^\top \mathbf{w}_v|\right),$$

where $\mathbf{w} \in \mathbf{R}^{nL}$ denotes the concatenation of $\mathbf{w}_v$ for all $v \in \mathcal{Y}$.

Theorem 3.1 *If the symmetric pairwise function $\bar{k} : \mathcal{Y} \times \mathcal{Y} \to \mathbf{R}$ defined by*

$$\bar{k}(u,v) = \begin{cases} k(v,v) & \textit{if } u = v, \\ -k(u,v) & \textit{otherwise,} \end{cases} \tag{3}$$

is positive semidefinite, then the function $\Omega(\mathbf{w})$ is convex.

Proof Given any two vectors $\mathbf{s}, \mathbf{t} \in \mathbf{R}^{nL}$ and any $\alpha \in [0,1]$, we have

$$\begin{array}{rl} & \alpha\Omega(\mathbf{s}) + (1-\alpha)\Omega(\mathbf{t}) - \Omega\big(\alpha\mathbf{s} + (1-\alpha)\mathbf{t}\big) \\ \geq & \frac{1}{2}\alpha(1-\alpha)[\sum_u k(u,u)\|\mathbf{s}_u - \mathbf{t}_u\|^2 + \sum_{u\neq v} k(u,v)(|\mathbf{s}_u^\top\mathbf{s}_v| + |\mathbf{t}_u^\top\mathbf{t}_v| - |\mathbf{s}_u^\top\mathbf{t}_v + \mathbf{t}_u^\top\mathbf{s}_v|)] \\ \geq & \frac{1}{2}\alpha(1-\alpha)[\sum_u k(u,u)\|\mathbf{s}_u - \mathbf{t}_u\|^2 - \sum_{u\neq v} k(u,v)|(\mathbf{s}_u - \mathbf{t}_u)^\top(\mathbf{s}_v - \mathbf{t}_v)|] \\ \geq & \frac{1}{2}\alpha(1-\alpha)[\sum_u k(u,u)\|\mathbf{s}_u - \mathbf{t}_u\|^2 - \sum_{u\neq v} k(u,v)\|\mathbf{s}_u - \mathbf{t}_u\|\|\mathbf{s}_v - \mathbf{t}_v\|] \\ \geq & 0. \end{array}$$

The second to the last inequality is based on the Cauchy-Schwarz inequality, and the last inequality holds because $\bar{k}$ is positive semidefinite[1]. This concludes the proof. ▌

Corollary 3.2 *If the symmetric pairwise function $\bar{k}$ defined in (3) is positive definite, then the objective function of (2) is strictly convex in $\mathbf{w}$, therefore it has a unique solution $\mathbf{w}^\star$.*

We find the following simple choice of the pairwise function k often perform very well in practice:

$$k(u,v) = k(v,u) = \begin{cases} |\mathcal{D}^+(v)| & \text{if } u = v, \\ \alpha & \text{if } u \in \mathcal{A}(v), \\ 0 & \text{else,} \end{cases} \tag{4}$$

where $\alpha > 0$ is a parameter. For many problems in practice, setting $\alpha = 1$ often give a positive definite $\bar{k}$, although it is certainly not always true. In any case, we can always reduce the value of α, or increase the diagonal values $k(v,v)$, to make $\bar{k}$ positive definite.

It is worth noticing that the choice of k in (4) implies $k(u,u) > k(v,v)$ whenever $u \in \mathcal{A}(v)$. This effectively encourages $\|\mathbf{w}_u\| < \|\mathbf{w}_v\|$ for $u \in \mathcal{A}(v)$. As shown in the classical theory on SVM [12], given a linear classifier $\mathbf{w}$, a small norm $\|\mathbf{w}\|$ corresponds to a large classification margin. Hence, the choice of k in (4) tends to give a larger classification margin at a higher-level classification. At the higher or more general levels, classifications are often relatively easier. It becomes increasingly more difficult as the categories become more specific. Thus, in general, from bottom to top, increased classification margins should be desirable.

[1]In fact, we only need $\bar{k}$ to be *copositive*, i.e., positive semidefinite on the set of nonnegative vectors.

3.2 Representer Theorem

Theorem 3.3 *If the pairwise function $\bar{k}$ is positive definite, then the solution to the optimization problem (2) admits a representation of the form $\mathbf{w}_v = \sum_{i=1}^m c_{vi}\mathbf{x}_i$ for any $v \in \mathcal{Y}$.*

Proof Assume $\{\mathbf{w}_v | v \in \mathcal{Y}\}$ is the solution to (2). Define a subspace $\mathcal{H} = \mathrm{span}\{\mathbf{x}_i | 1 \le i \le m\}$. Let $\mathcal{H}^\perp$ denote the orthogonal complement of $\mathcal{H}$. Thus, for each $v \in \mathcal{Y}$, we can decompose $\mathbf{w}_v$ such that $\mathbf{w}_v = \mathbf{s}_v + \mathbf{t}_v$ with $\mathbf{s}_v \in \mathcal{H}$ and $\mathbf{t}_v \in \mathcal{H}^\perp$. We then immediately have $\mathbf{w}_u^\top \mathbf{x}_i - \mathbf{w}_v^\top \mathbf{x}_i = \mathbf{s}_u^\top \mathbf{x}_i - \mathbf{s}_v^\top \mathbf{x}_i$. Hence, $\{\mathbf{s}_v | v \in \mathcal{Y}\}$ satisfies all constraints in (2). Moreover,

$$\begin{aligned}
2\Omega(\mathbf{w}) &= \textstyle\sum_u k(u,u)\|\mathbf{w}_u\|^2 + \sum_{u\neq v} k(u,v)|\mathbf{w}_u^\top \mathbf{w}_v| \\
&= \textstyle\sum_u k(u,u)(\|\mathbf{s}_u\|^2 + \|\mathbf{t}_u\|^2) + \sum_{u\neq v} k(u,v)|\mathbf{s}_u^\top \mathbf{s}_v + \mathbf{t}_u^\top \mathbf{t}_v| \\
&\ge \textstyle\sum_u k(u,u)\|\mathbf{s}_u\|^2 + \sum_{u\neq v} k(u,v)|\mathbf{s}_u^\top \mathbf{s}_v| + \sum_u k(u,u)\|\mathbf{t}_u\|^2 - \sum_{u\neq v} k(u,v)|\mathbf{t}_u^\top \mathbf{t}_v| \\
&\ge 2\Omega(\mathbf{s}) + (\textstyle\sum_u k(u,u)\|\mathbf{t}_u\|^2 - \sum_{u\neq v} k(u,v)|\mathbf{t}_u||\mathbf{t}_v|).
\end{aligned}$$

The second term in the last inequality is nonnegative when $\bar{k}$ is positive definite. Hence, $\Omega(\mathbf{w}) \ge \Omega(\mathbf{s})$. Since $\mathbf{w}$ is the solution, we should also have $\Omega(\mathbf{w}) \le \Omega(\mathbf{s})$. So $\Omega(\mathbf{w}) = \Omega(\mathbf{s})$. The equality holds if and only if $\mathbf{t}_v = 0$ for all $v \in \mathcal{Y}$. This concludes the proof. ∎

The representer theorem implies that our method can be considered in a more general reproducing kernel Hilbert space (RKHS) by choosing a problem specific reproducing kernel. Therefore more expressive nonlinear classifiers can be established [10].

4 A Regularized Dual Averaging Method

Establishing convexity of an optimization problem does not always mean that it can be readily solved by existing algorithms and available software. This is certainly the case for the problem (2). In particular, it cannot be easily transformed into any standard conic optimization form that lends itself to efficient interior-point methods. It does not fit any available special-purpose SVM solvers either. Here we develop a variant of the regularized dual averaging (RDA) method [9, 16] that is particularly suitable for solving problems like (2).

First, we transform the problem (2) into an equivalent unconstrained optimization problem

$$\begin{aligned}
\underset{\mathbf{w}\in\mathbf{R}^{nL}}{\text{minimize}} \quad J(\mathbf{w}) \;&\triangleq\; \tfrac{1}{2}\textstyle\sum_{v\in V} k(v,v)\|\mathbf{w}_v\|^2 + \sum_{v\in V}\sum_{u\in A(v)} k(u,v)\big|\mathbf{w}_u^T\mathbf{w}_v\big| \\
&\quad + \tfrac{C}{m}\textstyle\sum_{i=1}^m \max\big\{0, \max_{u\in S(v), v\in A^+(y_i)}\big\{1 - \mathbf{w}_v^T\mathbf{x}_i + \mathbf{w}_u^T\mathbf{x}_i\big\}\big\}\,.
\end{aligned} \tag{5}$$

In order to use the RDA method presented in Algorithm 4.1, we need to further decompose $J(\mathbf{w})$ into two separate terms, one is convex, and the other is a simple strongly convex function. More specifically, assume the matrix $\bar{k}$ (see Theorem 3.1) is positive definite, and let $\lambda_{\min} > 0$ be its smallest eigenvalue. Then we can split the objective function in (5) as $J(\mathbf{w}) = \phi(\mathbf{w}) + \Psi(\mathbf{w})$, with

$$\begin{aligned}
\phi(\mathbf{w}) &= \tfrac{1}{2}\textstyle\sum_{v\in\mathcal{Y}}\big(k(v,v) - \lambda_{\min}\big)\|\mathbf{w}_v\|^2 + \sum_{v\in V}\sum_{u\in A(v)} k(u,v)\big|\mathbf{w}_u^T\mathbf{w}_v\big| \\
&\quad + \tfrac{C}{m}\textstyle\sum_{i=1}^m \max\big\{0, \max_{u\in S(v), v\in A^+(y_i)}\big\{1 - \mathbf{w}_v^T\mathbf{x}_i + \mathbf{w}_u^T\mathbf{x}_i\big\}\big\}\,, \\
\Psi(\mathbf{w}) &= \tfrac{\lambda_{\min}}{2}\textstyle\sum_{v\in\mathcal{Y}}\|\mathbf{w}_v\|^2 \;=\; \tfrac{\lambda_{\min}}{2}\|\mathbf{w}\|^2.
\end{aligned}$$

This way, $\Psi(\mathbf{w})$ is strongly convex with modulus $\lambda_{\min}$, and the function $\phi(\mathbf{w})$ is still convex, since subtracting $\lambda_{\min}$ from the diagonals of $\bar{k}$ still gives a positive semidefinite pairwise function. Due to the simple form of $\Psi(\mathbf{w})$, the second step in the iterations of Algorithm 4.1 has a closed-form solution: $\mathbf{w}(t+1) = -(1/\lambda_{\min})\bar{\mathbf{g}}(t)$.

As shown in [16], this algorithm has a convergence rate $O(\ln t/t)$, which is much faster than $O(1/\sqrt{t})$ when using a plain subgradient method for nonsmooth convex optimization. This faster rate can also be achieved by using the Pegasos algorithm [11]. However, a distinct feature of Algorithm 4.1 is that it provides both an upper bound and a lower bound on the optimal value at each iteration. Therefore we have an effective stopping criterion to produce an ϵ-approximate solution. Updating the upper bounds $\hat{J}(t)$ in Algorithm 4.1 is straightforward, so we only need to explain the lower bounds $\check{J}(t)$. Since ϕ is convex and $\mathbf{g}(t)$ is a subgradient of ϕ at $\mathbf{w}(t)$, we have

$$J(\mathbf{w}) \ge \tfrac{1}{t}\textstyle\sum_{\tau=1}^t \big[\phi(\mathbf{w}(\tau)) + \mathbf{g}(\tau)^\top(\mathbf{w} - \mathbf{w}(\tau))\big] + \Psi(\mathbf{w}), \quad \forall\, \mathbf{w} \in \mathbf{R}^{nL}.$$

Algorithm 4.1 Regularized Dual Averaging (RDA) Method with Bound on Optimality Gap

```
input: training examples $\{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$, constant $C > 0$, and accuracy $\epsilon > 0$
initialize: $t = 1$, $\mathbf{w}(1) = 0$, $\mathbf{w}^* = 0$, $\bar{\mathbf{g}}(0) = 0$, $\delta(0) = 0$, $\hat{J}(1) = C$, $\check{J}(1) = 0$
  repeat
      • Compute a subgradient $\mathbf{g}(t) \in \partial\phi(\mathbf{w}(t))$, and update $\bar{\mathbf{g}}(t) = \frac{t-1}{t}\bar{\mathbf{g}}(t-1) + \frac{1}{t}\mathbf{g}(t)$.
      • Compute the next weight vector: $\mathbf{w}(t+1) = \mathrm{argmin}_{\mathbf{w}\in\mathbf{R}^{nL}} \{\bar{\mathbf{g}}(t)^\top \mathbf{w} + \Psi(\mathbf{w})\}$.
      • Update the upper bound: If $J(\mathbf{w}(t+1)) < \hat{J}(t)$, then $\mathbf{w}^* = \mathbf{w}(t+1)$, $\hat{J}(t+1) = J(\mathbf{w}^*)$.
      • Update the lower bound:
            $\delta(t) = \frac{t-1}{t}\delta(t-1) + \frac{1}{t}\big(\phi(\mathbf{w}(t)) - \mathbf{g}(t)^\top \mathbf{w}(t)\big)$,
            $\check{J}(t+1) = \max\{\check{J}(t),\ \delta(t) + \bar{\mathbf{g}}(t)^\top \mathbf{w}(t+1) + \Psi(\mathbf{w}(t+1))\}$.
  until $\hat{J}(t+1) - \check{J}(t+1) \le \epsilon$
  return $\mathbf{w}^*$
```

Taking the minimum on both sides of the previous inequality, we have

$$\begin{aligned}
\min_{\mathbf{w}\in\mathbf{R}^{nL}} J(\mathbf{w}) &\geq \min_{\mathbf{w}\in\mathbf{R}^{nL}} \left\{ \frac{1}{t}\sum_{\tau=1}^{t} \big[\phi(\mathbf{w}(\tau)) + \mathbf{g}(\tau)^\top(\mathbf{w} - \mathbf{w}(\tau))\big] + \Psi(\mathbf{w}) \right\} \\
&= \frac{1}{t}\sum_{\tau=1}^{t} \big[\phi(\mathbf{w}(\tau)) - \mathbf{g}(\tau)^\top \mathbf{w}(\tau)\big] + \min_{\mathbf{w}\in\mathbf{R}^{nL}} \{\bar{\mathbf{g}}(t)^\top \mathbf{w} + \Psi(\mathbf{w})\} \\
&= \delta(t) + \bar{\mathbf{g}}(t)^\top \mathbf{w}(t+1) + \Psi(\mathbf{w}(t+1)).
\end{aligned}$$

The last line above is precisely what is used to update the lower bound $\check{J}(t)$ in Algorithm 4.1.

5 Preliminary Experiments

We evaluated our method on several classification tasks derived from the popular text categorization benchmark called RCV1-v2/LYRL2004 [7]. It contains 23,149 training documents and 781,265 test documents, for a total of 804,414 documents. The documents have been tokenized, stopworded and stemmed to 47,236 unique tokens and represented as L2-normalized log TF-IDF vectors. The top categories MCAT, CCAT and ECAT were used to form three classification tasks. We did not consider GCAT because its subcategories are flat rather than being organized hierarchically. In each classification task, we remove the top categories that do not have any subcategory to emphasize the effect of hierarchy. In addition, the documents without unique labeling path or leaf node label were discarded. Then we further excluded C15 from CCAT since it is significantly much larger than other categories. E13 and E14 were also excluded from ECAT since their subcategories contain too few training examples. We also evaluated our method on the 20-News dataset which contains 18,846 documents and 61,188 unique tokens [2]. We computed the L2-normalized log TF-IDF vectors instead of using the default term frequency based representation. Then we took the two topics *comp* and *rec* which contains 8,820 documents and the associated hierarchy in our evaluation.

We compared our method with the following methods:

- Flat multiclass SVM. This is the formulation proposed in [2], using only the leaf categories.
- Hierarchical multiclass SVM. This is (2) with $k(v, v) = 1$ and $k(u, v) = 0$ if $u \neq v$.
- Hierarchical multitask SVM. This is a multitask version of SVM proposed in [5].
- Hierarchical SVM (path loss). This is the method in [1] with a tree-induced losses.
- Hierarchical SVM (0/1 loss). This is the method in [1] adapted with zero-one losses.

All methods were trained with $C = 1$, which is commonly used in text classification (e.g., [7, 1, 14]). We also tried different values of C varying from 1 to 100 and did not notice any significant difference in classification performance. For our method of orthogonal transfer, we use the parameters in (4) and simply setting $\alpha = 1$. We used Algorithm 4.1 to solve all the above formulations.

We collect the average performance over 50 rounds of random splitting of the training/testing datasets. For RCV1-v2/LYRL2004, the training/testing split ratio in each round was the same as

[2]See `http://people.csail.mit.edu/jrennie/20Newsgroups/`.

Table 1: Average classification error ($0/1$ loss) and standard deviation over 50 random splittings.

METHODS	MACT	CCAT	ECAT	20-NEWS
FLAT MULTICLASS SVM	$5.26(\pm 0.20)$	$21.49(\pm 0.27)$	$11.85(\pm 0.29)$	$11.50(\pm 0.50)$
HIER. MULTICLASS SVM	$4.87(\pm 0.18)$	$21.48(\pm 0.31)$	$12.09(\pm 0.34)$	$11.37(\pm 0.49)$
HIER. MULTITASK SVM	$4.73(\pm 0.18)$	$21.99(\pm 0.32)$	$12.05(\pm 0.33)$	$11.36(\pm 0.48)$
HIER. SVM (PATH LOSS)	$13.55(\pm 0.60)$	$26.48(\pm 0.42)$	$15.40(\pm 0.43)$	$33.22(\pm 1.14)$
HIER. SVM ($0/1$ LOSS)	$6.65(\pm 0.22)$	$22.21(\pm 0.31)$	$13.01(\pm 0.32)$	$11.95(\pm 0.54)$
ORTHOGONAL TRANSFER	$\mathbf{3.03}(\pm \mathbf{0.13})$	$\mathbf{17.53}(\pm \mathbf{0.55})$	$\mathbf{10.01}(\pm \mathbf{0.28})$	$\mathbf{11.19}(\pm \mathbf{0.46})$

the default setting. For 20-News, we randomly sampled 60% data for training and used the remaining for testing. The classification errors based on zero-one loss for all the methods are summarized in Tables 1. Our orthogonal transfer method clearly outperformed others on every task from RCV1-v2/LYRL2004, and on 20-News it was just slightly better than others approaches. A possible explanation is that the hierarchy of RCV1-v2/LYRL2004 is more meaningful than that of 20-News.

6 Conclusions

We proposed a novel method called orthogonal transfer for hierarchical classification that specifically tackles the difficulty of classifying similar classes in the lower levels of the category hierarchy. We presented a convex optimization formulation for the problem and devised an efficient dual averaging method for solving it. Preliminary empirical evaluations show that our method can effectively exploit the hierarchical structure and is able to produce improved classification accuracy. As a future work, we are very interested in analyzing orthogonal transfer from a learning theory perspective.

References

[1] L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proc. of 13th ACM Internat. Conf. on Information and Knowledge Management*, pages 78–87, 2004.

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

[3] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *Proceedings of the 21st International Conference on Machine Learning*, pages 27–34, 2004.

[4] S. T. Dumais and H. Chen. Hierarchical classification of web content. In *Proceedings of SIGIR'00*, pages 256–263, 2000.

[5] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[6] D. Koller and M. Sahami. Hierarchically classifying docuemnts using very few words. In *Proceedings of the 14th International Conference on Machine Learning*, pages 171–178, 1997.

[7] D. D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[8] A. K. McCallum, R. Rosenfeld, T. M. Mitchell, and A. Y. Ng. Improving text classification by shrinkage in a hierarchy of classes. In *Proc. of 15th Internat. Conf. on Machine Learning*, pages 359–367, 1998.

[9] Yu. Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical Programming*, 120:221–259, 2009.

[10] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[11] S. Shalev-Shwartz, Y. Singer, , and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *Proc. of 24th Internat. conf. on Machine learning (ICML)*, pages 807–814, 2007.

[12] V. N. Vapnik. *Statistical learning theory*. John Wiley & Sons, New York, 1998.

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

[14] K. Weinberger and O. Chapelle. Large margin taxonomy embedding with an application to document categorization. In *Advances in Neural Information Processing Systems 21*, pages 1737–1744, 2008.

[15] J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proc. of 6th European Symposium on Artificial Neural Networks*, pages 219–224, 1999.

[16] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11:2543–2596, 2010.